SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF ELECTRICAL AND ELECTRONICS

DEPARTMENT OF ELECTRONICS AND COMMMUNICATION ENGINEERING

Digital Signal Processing

UNIT - I

SECA1501 - DISCRETE TIME SIGNALS AND SYSTEMS

UNIT 1 DISCRETE TIME SIGNALS AND SYSTEMS

Introduction to DSP – Basic elements of DSP-Representation, Sampling theorem - Aliasing effect, Characterization and Classifications of Discrete Time (DT) signals, Operations on DT signals , Convolution, Advantages of DSP over ASP , Classification of DT systems , properties of Discrete time systems-Linearity-Time invariance- causality -stability -Linear time Invariant systems-The Z transform- Inverse Z transform-System transfer Function

Introduction to DSP

DSP manipulates different types of signals with the intention of filtering, measuring, or compressing and producing analog signals. Analog signals differ by taking information and translating it into electric pulses of varying amplitude, whereas digital signal information is translated into binary format where each bit of data is represented by two distinguishable amplitudes. Another noticeable difference is that analog signals can be represented as sine waves and digital signals are represented as square waves. DSP can be found in almost any field, whether it's oil processing, sound reproduction, radar and sonar, medical image processing, or telecommunications-- essentially any application in which signals are being compressed and reproduced.

Analog Signal

Digital Signal

So what exactly is digital signal processing? The digital signal process takes signals like audio, voice, video, temperature, or pressure that has already been digitized and then manipulates them mathematically. This information can then be represented as discrete time, discrete frequency, or other discrete forms so that the information can be digitally processed. An analog-to-digital converter is needed in the real world to take analog signals (sound, light, pressure, or temperature) and convert them into 0's and 1's for a digital format.

Continuous Time signal – If the signal is defined over continuous-time, then the signal is a continuous-time signal.

Ex: Sinusoidal signal, Voice signal, Rectangular pulse function

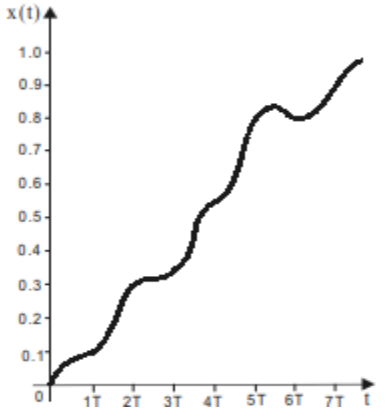


Fig 1 Continuous Time signal

Discrete Signal and Discrete Time Signal:

The discrete signal is a function of a discrete independent variable. The independent variable is divided into uniform intervals and each interval is represented by an integer. The letter "n" is used to denote the independent variable. The discrete or digital signal is denoted by x(n).

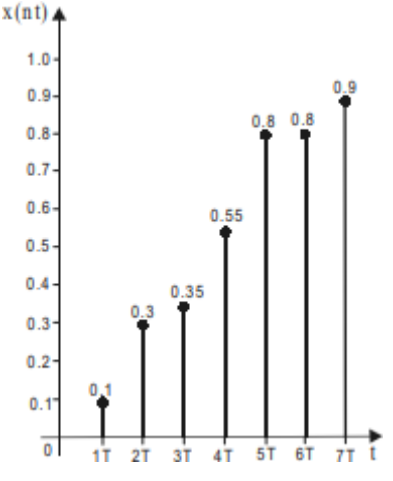


Fig 2: Discrete Time Signal

Digital Signal: The signals that are discrete in time and quantized in amplitude are called digital signal. The term "digital signal" applies to the transmission of a sequence of values of a discrete-time signal in the form of some digits in the encoded form.

Representation of Discrete Time Signals

1. **Functional representation**

In functional representation, the signal is represented as a mathematical equation, as shown in the following example.

$$x(n) = \begin{cases} -0.5 & ; \; n = -2 \\ 1.0 & ; \; n = -1 \\ -1.0 & ; \; n = 0 \\ 0.6 & ; \; n = 1 \\ 1.2 & ; \; n = 2 \\ 1.5 & ; \; n = 3 \\ 0 & ; \; \text{other } n \end{cases}$$

2. Graphical representation

In graphical representation, the signal is represented in a two-dimensional plane. The independent variable is represented in the horizontal axis and the value of the signal is represented in the vertical axis as shown below

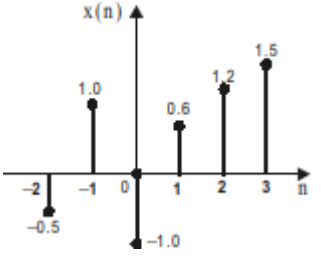


Fig 3: Discrete Time Signal

3. Tabular representation

In tabular representation, two rows of a table are used to represent a discrete time signal. In the first row, the independent variable "n" is tabulated and in the second row the value of the signal for each value of "n" are tabulated as shown in the following table I.

n	.....	–2	-1	0	1	2	3	.....
x(n)	.....	–0.5	1.0	–1.0	0.6	1.2	1.5	.....

Sequence representation

In sequence representation, the discrete time signal is represented as a one-dimensional array as shown in the following examples.

An infinite duration discrete time signal with the time origin, n = 0, indicated by the symbol - is represented as, x(n) = { ..... –0.5, 1.0, –1.0, 0.6, 1.2, 1.5, }

A symbol represented with ↑ shows that the signal is starting at the instant n=0.

An infinite duration discrete time signal that satisfies the condition x(n) = 0 for n < 0 is represented as,

x(n) = { –1.0, 0.6, 1.2, 1.5, ... } or x(n) = {–1.0, 0.6, 1.2, 1.5, . }

A finite duration discrete time signal with the time origin, n = 0, indicated by the symbol - is represented as, x(n) = { – 0.5, 1.0, –1.0, 0.6, 1.2, 1.5 }

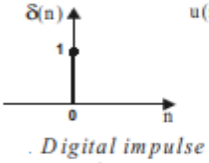
A finite duration discrete time signal that satisfies the condition x(n) = 0 for n < 0 is represented as,

x(n) = { –1.0, –0.6, 1.2, 1.5 } or x(n) = { –1.0, 0.6, 1.2, 1.5}

Standard Discrete Time Signals

1. Digital impulse signal or unit sample sequence

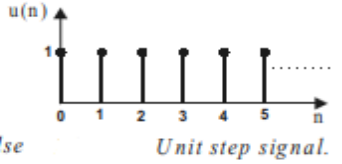
Impulse signal, $\delta(n) = 1 \;;\; n = 0$
$= 0 \;;\; n \neq 0$



Digital impulse signal.

2. Unit step signal

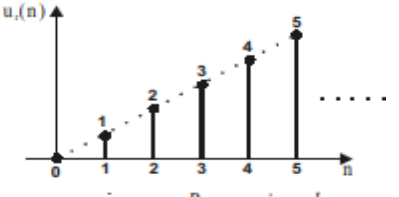
Unit step signal, $u(n) = 1 \;;\; n \geq 0$
$= 0 \;;\; n < 0$



Unit step signal.

3. Ramp signal

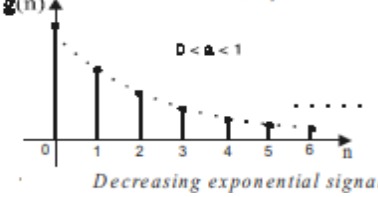
Ramp signal, $u_r(n) = n \;;\; n \geq 0$
$= 0 \;;\; n < 0$



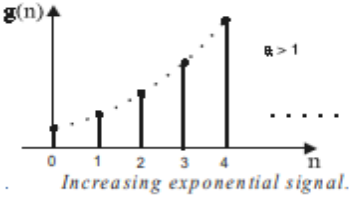
Ramp signal.

4. Exponential signal

Exponential signal, $g(n) = a^n \;;\; n \geq 0$
$= 0 \;;\; n < 0$



Decreasing exponential signal.



Increasing exponential signal.

Fig 4: Standard Discrete Time Signals

Classification of Discrete Time Signals

The discrete time signals are classified depending on their characteristics. Some ways of classifying discrete time signals are,

1. **Deterministic and nondeterministic signals**
2. **Periodic and aperiodic signals**
3. **Symmetric and antisymmetric signals**
4. **Energy and power signals**
5. **Causal and noncausal signals**

Deterministic and Nondeterministic Signals

The signals that can be completely specified by mathematical equations are called deterministic signals. The step, ramp, exponential and sinusoidal signals are examples of deterministic signals. The signals whose characteristics are random in nature are called nondeterministic signals. The noise signals from various sources are best examples of nondeterministic signals.

Periodic and Aperiodic Signals

When a discrete time signal x(n), satisfies the condition x(n + N) = x(n) for integer values of N, then the discrete time signal x(n) is called periodic signal. Here N is the number of samples of a period.

i.e, if, x(n + N) = x(n), for all n, then x(n) is periodic

The smallest value of N for which the above equation is true is called fundamental period. If there is no value of N that satisfies the above equation, then x(n) is called aperiodic or nonperiodic signal. When N is the fundamental period, the periodic signals will also satisfy the condition x(n + kN) = x(n), where k is an integer. The periodic signals are power signals. The discrete time sinusoidal and complex exponential signals are periodic signals when their fundamental frequency, f_0 is a rational number.

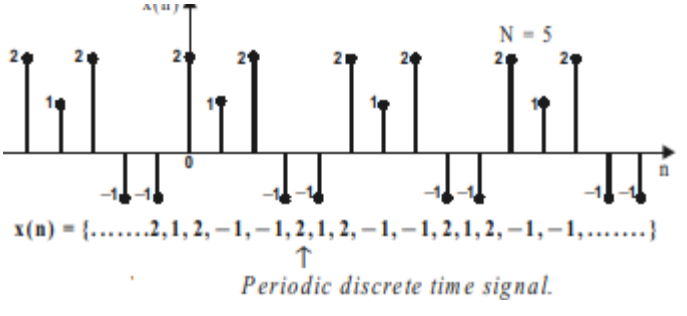


Fig 5. Periodic Discrete Time Signals

Symmetric (Even) and Antisymmetric (Odd) Signals

The discrete time signals may exhibit symmetry or antisymmetry with respect to n = 0. When a discrete time signal exhibits symmetry with respect to n = 0 then it is called an even signal. Therefore, the even signal satisfies the condition, x(n)=x(-n)

When a discrete time signal exhibits antisymmetry with respect to n = 0, then it is called an odd signal. Therefore the odd signal satisfies the condition,

$$x(-n) = -x(n)$$

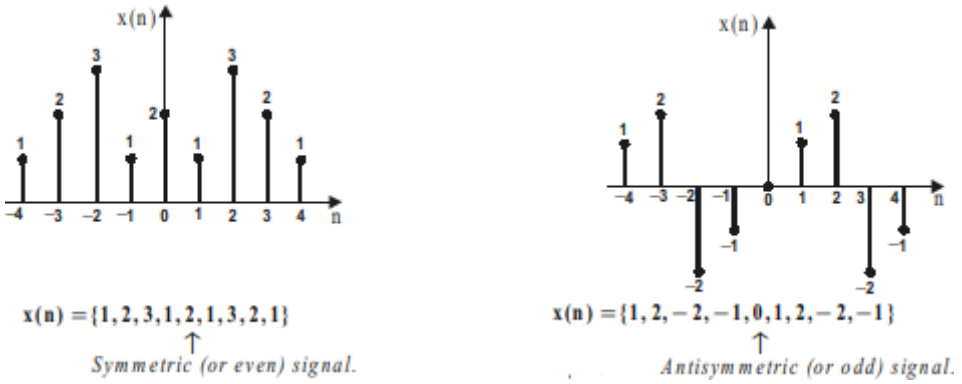


Fig 6. Symmetric and antisymmetric Discrete Time Signals

Energy and Power Signals

The energy E of a discrete time signal x(n) is defined as,

$$\text{Energy, } E = \sum_{n=-\infty}^{\infty} |x(n)|^2$$

The energy of a signal may be finite or infinite, and can be applied to complex valued and real valued signals. If energy E of a discrete time signal is finite and nonzero, then the discrete time signal is called an energy signal. The exponential signals are examples of energy signals. The average power of a discrete time signal x(n) is defined as,

$$\text{Power, } P = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} |x(n)|^2$$

If power P of a discrete time signal is finite and nonzero, then the discrete time signal is called a power signal. The periodic signals are examples of power signals. For energy signals, the energy will be finite and average power will be zero. For power signals the average power is finite and energy will be infinite.

$$\text{For energy signal, } 0 < E < \infty \text{ and } P = 0$$

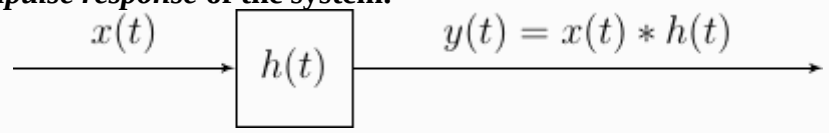
$$\text{For power signal, } 0 < P < \infty \text{ and } E = \infty$$

Causal, Noncausal and Anticausal signals

A discrete time signal is said to be causal, if it is defined for n ³ 0. Therefore if x(n) is causal, then x(n) = 0 for n < 0. A discrete time signal is said to be noncausal, if it is defined for either n ≤ 0, or for both n ≤ 0 and n > 0. Therefore if x(n) is noncausal, then x(n) ≠ 0 for n < 0. A noncausal signal can be converted to causal signal by multiplying the noncausal signal by a unit step signal, u(n). When a noncausal discrete time signal is defined only for n ≤ 0, it is called an anticausal signal.

Convolution

A linear *shift invariant* system can be described as convolution of the input signal. The kernel used in the convolution is the *impulse response* of the system.



A (continuous time) Shift Invariant Linear System is characterized with its impulse response.

A proof for this fact is easiest for discrete time signals. The proof for discrete time signals is left as an exerise for the reader. Here we consider continuous time signals.

Let xx be the input signal to a linear system LL and let the output be y=Lxy=Lx. We can write xx as an integration (summation) of shifted pulses:

$$x(t)=\int_{-\infty}^{\infty}x(u)\delta(u-t)dux(t)=\int_{-\infty}^{\infty}x(u)\delta(u-t)du$$

Because δ(x)=δ(−x)δ(x)=δ(−x) we can also write:

$$x(t)=\int_{-\infty}^{\infty}x(u)\delta(t-u)du=\int_{-\infty}^{\infty}x(u)\delta u(t)dux(t)=\int_{-\infty}^{\infty}x(u)\delta(t-u)du=\int_{-\infty}^{\infty}x(u)\delta u(t)du$$

where δu(t)δu(t) is the function δδ shifted to the left over uu. Now look at LxLx. Because of the linearity of LL we may write:

$$(Lx)(t)=\int_{-\infty}^{\infty}x(u)(L\delta u)(t)du(Lx)(t)=\int_{-\infty}^{\infty}x(u)(L\delta u)(t)du$$

Shift invariance of the operator implies that (Lδu)=(Lδ)u(Lδu)=(Lδ)u, i.e. first shifting and then applying the operator is the same as first applying the operator and then shift.

Obviously LδLδ is the *pulse response* of the linear system, let's call it the function hh, then we get:

$$(Lx)(t)=y(t)=\int_{-\infty}^{\infty}x(u)h(t-u)du(Lx)(t)=y(t)=\int_{-\infty}^{\infty}x(u)h(t-u)du$$

or equivalently:

$$y=x*h,\ y=x*h$$

the output of a shift invariant system is given by the convolution of the input signal with the impulse response function of the system. In the signal processing literature it is common to write:

$$y(t)=x(t)*h(t)y(t)=x(t)*h(t)$$

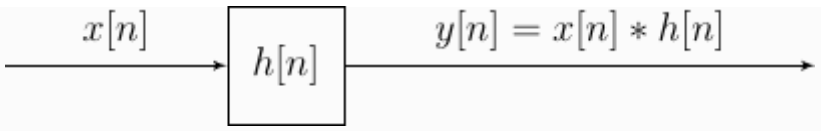
Although this is a bit sloppy notation (for a mathematician this looks like an expression involving real numbers not functions) it is used a lot and even in some cases it helps to make clear what the functions involved.

Consider the case of discrete time signals. Let x[n]x[n] be the input signal to a linear LTI system

that is characterized with its impulse response h[n]h[n]. The output signal then is given by:

$$y[n]=x[n]*h[n]y[n]=x[n]*h[n]$$

So although mathematically quite sloppy this notation allows clear distinction between continuous time and discrete time systems.



A (discrete time) Shift Invariant Linear System is characterized with its impulse response.

Linear Convolution :

An arbitrary input signal *x(n)* in to a weighted sum of impulses, We are now ready to determine the

$$y(n, k) \equiv h(n, k) = \mathcal{T}[\delta(n-k)]$$

response of any relaxed linear system to any Input signal. First, we denote the response y(n,k) of the system to the input unit Sample sequence at *n* = k by the special symbol *h(n, k),* $-\infty<k<\infty$. T h a t is,

if the input is the arbitrary signal x(n) that is expressed as a sum of weighted impulses, that is.

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

then the response of the system to *x(n)* is the corresponding sum of weighted outputs, that is,

$$y(n) = \mathcal{T}[x(n)] = \mathcal{T}\left[\sum_{k=-\infty}^{\infty} x(k)\delta(n-k)\right]$$

$$= \sum_{k=-\infty}^{\infty} x(k)\mathcal{T}[\delta(n-k)]$$

$$= \sum_{k=-\infty}^{\infty} x(k)h(n, k)$$

clearly, the above equation follows from the superposition property of linear systems, and is know n as the *superposition summation.*th en by the time-invariance property , the responseof the system to the delayed unit sample sequence δ(n - *k)* is

$$h(n-k) = \mathcal{T}[\delta(n-k)]$$

Consequently , the *superposition summation* formula in reduces to

$$y(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k)$$

The above formula gives the response *y(n)* of the LTI system as a function of the input signal *x (n)* and the unit sample (impulse) response *h(n)* is called a *convolution sum.*

To summarize, the process of computing the convolution between *x (k)* and *h(k)* involves the following four steps.

1. Folding. **Fold *h(k)* about *k* = 0 to obtain *h (- k)* .**

2. Shifting, **Shift *h (—k)* by n0 to the right (left) if n0 is positive (negative), to obtain *h(*n0—k).**

3. Multiplication. **Multiply *x (k)* by *h(*n0— k) to obtain the product sequence *v* n0*(k) = x (k) h(*n0— k).**

4. Summation. **Sum all the values o f the product sequence *v*n0*(k*) to obtain the value of the output at time *n* = n0.**

Example:

The impulse response of a linear time-invariant system is Determine the response of the system

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

to

the input signal

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

Solution : We shall compute the convolution according to its formula. But we shall use graphs of the sequences to aid us in the computation. In Fig. below we illustrate the input signal sequence *x(k)* and the impulse response *h{k)* of the system, using *k* as the time index. The first step in the computation of the convolution sum is to fold *h(k).* The folded sequence *h(-k)* is illustrated inconsequent figs . Now we can compute the output at *n* = 0. according to the convolution formula which is

$$y(0) = \sum_{k=-\infty}^{\infty} x(k)h(-k)$$

Since the shift *n* = 0, we use *h(—k)* directly without shifting it. The product

$$v_0(k) \equiv x(k)h(-k)$$

sequence We continue the computation by evaluating the response of the system

at *n* = 1.

$$y(1) = \sum_{k=-\infty}^{\infty} x(k)h(1-k)$$

Finally, the sum of all the values in the product sequence yields

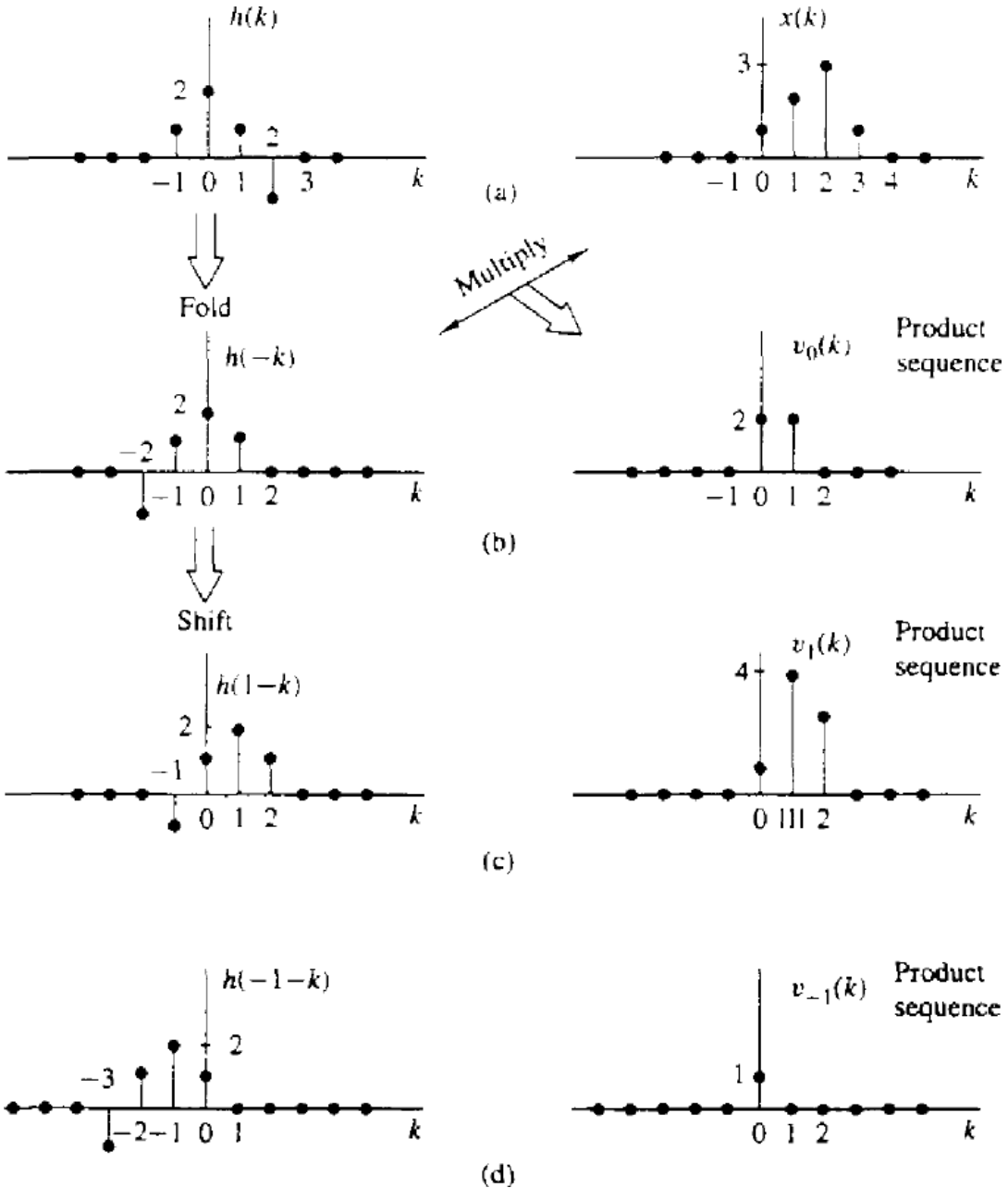
$$y(1) = \sum_{k=-\infty}^{\infty} v_1(k) = 8$$

In a similar manner, we can obtain y(2) by shifting *h (- k)* two units to the right. And y(2) = 8.

Then y(3) = 3. y(4) = - 2 , y(5) = -1 .For *n* >5, we find that y(n) = 0 because the product sequences contain all zeros.

Next we wish to evaluate y(n) for *n* < 0. We begin with *n =-1.Then*

$$y(-1) = \sum_{k=-\infty}^{\infty} x(k)h(-1-k)$$



$$y(0) = \sum_{k=-\infty}^{\infty} v_0(k) = 4$$

$$y(-1) = 1$$

Finally, summing over the values of the product sequence, we obtain

$$y(n) = 0 \qquad \text{for } n \leq -2$$

Now we have the entire response of the system for $-\infty < n < \infty$. which we summarize below as

$$y(n) = \{\ldots, 0, 0, \underset{\uparrow}{1}, 4, 8, 8, 3, -2, -1, 0, 0, \ldots\}$$

Properties of Convolution:

1- Commutative law :

$$x(n) * h(n) = h(n) * x(n)$$

2- Associative law :

$$[x(n) * h_1(n)] * h_2(n) = x(n) * [h_1(n) * h_2(n)]$$

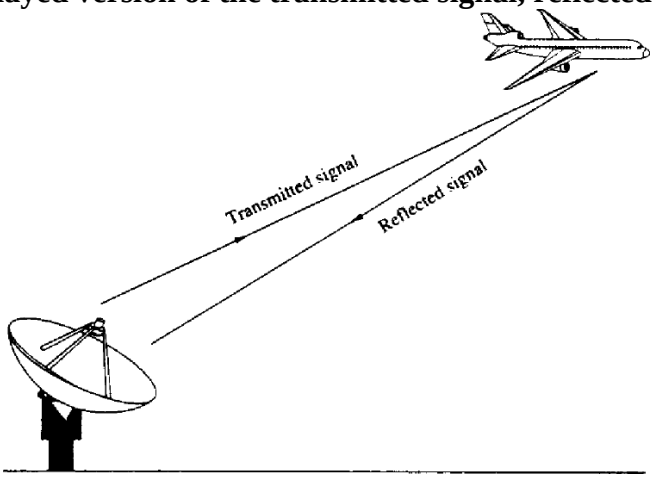
3-Distributive law :

$$x(n) * [h_1(n) + h_2(n)] = x(n) * h_1(n) + x(n) * h_2(n)$$

CORRELATION OF DISCRETE-TIME SIGNALS:

A mathematical operation that closely resembles convolution is correlation .Just as in the case of convolution , two signal sequences are involved in correlation. correlation between the two signals is to measure the degree to which the two signals are similar and thus to extract some information that depends to a large extent on the application. Correlation o f signals is often encountered in radar, sonar, digital communications, geology, an do the rare as in science and engineering .

Let us suppose that we have two signal sequences x(n) and y(n) that we wish to compare. In radar and active sonar applications. x(n) can represent the sampled version of the transmitted signal and y{n) can represent the sampled version of the received signal at the output of the analog -to -digital (A /D) converter. If a target is p resent in the space being searched by the radar or sonar, the received signal y(n) consists of a delayed version of the transmitted signal, reflected from the target.



This comparison process is performed by means of the correlation operation of 2 different

types.

Cross-correlation and Autocorrelation Sequences :
Suppose that we have two real signal sequences *x(n)* and *y(n)* each of which has finite energy. T he *cross-correlation* o f *x(n)* and y(n) is a sequence *rxy(l),* which is defined as

$$r_{xy}(l) = \sum_{n=-\infty}^{\infty} x(n)y(n-l) \qquad l = 0, \pm 1, \pm 2, \ldots$$

or, equivalently , as

$$r_{xy}(l) = \sum_{n=-\infty}^{\infty} x(n+l)y(n) \qquad l = 0, \pm 1, \pm 2, \ldots$$

The index l is the (time) shift (or *lag)* parameter and the subscripts *x y* on the cross-correlation se quence *rxy(l),* indicate the sequences being correlated .If we reverse the roles of x(n) an d y(n) and there fore reverse the order of the indices xy. we obtain the cross-correlation sequence

$$r_{yx}(l) = \sum_{n=-\infty}^{\infty} y(n)x(n-l)$$

or, equivalently ,

$$r_{yx}(l) = \sum_{n=-\infty}^{\infty} y(n+l)x(n)$$

By comparing the above 4 equations we conclude that

$$r_{xy}(l) = r_{yx}(-l)$$

Hence , *ryx(l*) provides exactly the same information as *rxy(l),*with respect to the similarity of *x (n)* to y(n).

Example:
Determine the cross-correlation sequence *rxy(l)* of the sequences

$$x(n) = \{\ldots 0, 0, 2, -1, 3, 7, 1, 2, -3, 0, 0, \ldots\}$$
$$\uparrow$$
$$y(n) = \{\ldots 0, 0, 1, -1, 2, -2, 4, 1, -2, 5, 0, 0, \ldots\}$$
$$\uparrow$$

Solution : Let us use the definition of cross-correlation to compute *rxy(l).* For *I* = 0 w e have

$$v_0(n) = \{\ldots, 0, 0, 2, 1, 6, -14, 4, 2, 6, 0, 0, \ldots\}$$
$$\uparrow$$

$$r_{xy}(0) = 7$$

$$r_{xy}(0) = \sum_{n=-\infty}^{\infty} x(n)y(n)$$

The product sequence v0*(n)* =*x (n) y(n)* is 1

For *I* > 0, we simply shift y(n) to the right relative to x(n) hy l units, compute the product sequence *vl(n)* = x(n)y(n — *I),* and finally, sum over all values o f the product sequence. Thus we obtain

$$r_{xy}(1) = 13, \quad r_{xy}(2) = -18, \quad r_{xy}(3) = 16, \quad r_{xy}(4) = -7$$

$$r_{xy}(5) = 5, \quad r_{xy}(6) = -3, \quad r_{xy}(l) = 0, \quad l \geq 7$$

For l< 0, we shift *y(n)* to the left relative to x(n) by l units, compute the product sequence vl(n) = x(n)y(n — *I),* and sum over all values of the product sequence. Thus we obtain the values of the cross-correlation sequence

$$r_{xy}(-1) = 0, \quad r_{xy}(-2) = 33, \quad r_{xy}(-3) = -14, \quad r_{xy}(-4) = 36$$

$$r_{xy}(-5) = 19, \quad r_{xy}(-6) = -9, \quad r_{xy}(-7) = 10, \quad r_{xy}(l) = 0, \ l \leq -8$$

Therefore, the cross-correlation sequence of *x{n)* and y(n) is

$$r_{xy}(l) = \{10, -9, 19, 36, -14, 33, 0, \underset{\uparrow}{7}, 13, -18, 16, -7, 5, -3\}$$

Then the convolution o f *x(n)* with y (—*n)* yields the cross-correlation *rxy(l)* that is,

$$r_{xy}(l) = x(l) * y(-l)$$

Autocorrelation*:*

when *y(n) = x(n),* we have the *autocorrelation* of x(n),which is defined as the sequence

$$r_{xx}(l) = \sum_{n=-\infty}^{\infty} x(n)x(n-l)$$

or, equivalently, as

$$r_{xx}(l) = \sum_{n=-\infty}^{\infty} x(n+l)x(n)$$

For finite-duration sequences,

$$r_{xy}(l) = \sum_{n=i}^{N-|k|-1} x(n)y(n-l)$$

and

$$r_{xx}(l) = \sum_{n=i}^{N-|k|-1} x(n)x(n-l)$$

where *i = l, k* = 0 for l> 0, and i = 0, *k = l* for l < 0.

Properties of the Autocorrelation and Cross correlation Sequences :

1- The cross-correlation sequence satisfies the condition that

$$|r_{xy}(l)| \le \sqrt{r_{xx}(0)r_{yy}(0)} = \sqrt{E_x E_y}$$

when y(n) = *x (n*), reduces to

$$|r_{xx}(l)| \le r_{xx}(0) = E_x$$

2- Th e normalized auto correlation sequence is defined as

$$\rho_{xx}(l) = \frac{r_{xx}(l)}{r_{xx}(0)}$$

Similarly, we define the normalized cross-correlation sequence

$$\rho_{xy}(l) = \frac{r_{xy}(l)}{\sqrt{r_{xx}(0)r_{yy}(0)}}$$

Now *ρxx{l)*\ <1 and *ρxy{l)*\ < 1, and hence these sequences are independent of

signal scaling. 3-the cross-correlation sequence satisfies the property

$$r_{xy}(l) = r_{yx}(-l)$$

the autocorrelation sequence satisfies the property Hence the auto correlation function is an even function. $r_{xx}(l) = r_{xx}(-l)$

The One-sided z-Transform:

The one-sided or unilateral z-transform of a signal x(n) is defined by $X(z)=\sum x^{n} z^{-n}$

Properties:

1. **It does not contain information about the signal x(n) for negative values of time.**
2. **It is unique only for causal signals.**
3. **The one-sided z-transform $X^{+}(z)$ of x(n) is identical to the two-sided z-transform of the signal x(n)u(n).**

Circular Convolution

The Circular Convolution property states that if DFT x1(n) X1(k) And N Then x1(n) N DFT x2(n) X2(k) Then N DFT x2(n) x1(k) x2(k) N It means that circular convolution of x1(n) & x2(n) is equal to multiplication of their DFTs. Thus circular convolution of two periodic discrete signal with period N is given by N-1 y(m) = ∑ x1 (n) x2 (m-n)N …….(4) n=0 Multiplication of two sequences in time domain is called as Linear convolution while Multiplication of two sequences in frequency domain is called as circular convolution. Results of both are totally different but are related with each other. There are two different methods are used to calculate circular convolution 1) Graphical representation form 2) Matrix approach

Concentric Circle Method

Let x1(n)x1(n) and x2(n)x2(n) be two given sequences. The steps followed for circular convolution of x1(n)x1(n) and x2(n)x2(n) are

- **Take two concentric circles. Plot N samples of x1(n)x1(n) on the circumference of the outer circle maintainingequaldistancesuccessivepointsmaintainingequaldistancesuccessivepoints i n anti-clockwise direction.**
- **For plotting x2(n)x2(n), plot N samples of x2(n)x2(n) in clockwise direction on the inner circle, starting sample placed at the same point as 0^{th} sample of x1(n)x1(n)**
- **Multiply corresponding samples on the two circles and add them to get output.**
- **Rotate the inner circle anti-clockwise with one sample at a time.**

Matrix Multiplication Method

Matrix method represents the two given sequence x1(n)x1(n) and x2(n)x2(n) in matrix form.

- **One of the given sequences is repeated via circular shift of one sample at a time to form a N X N matrix.**
- **The other sequence is represented as column matrix.**
- **The multiplication of two matrices give the result of circular convolution.**

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF ELECTRICAL AND ELECTRONICS

DEPARTMENT OF ELECTRONICS AND COMMMUNICATION ENGINEERING

Digital Signal Processing

UNIT - 2

SECA1501 - DISCRETE FOURIER TRANSFORM (DFT) AND FAST FOURIER TRANSFORM (FFT)

UNIT 2 DISCRETE FOURIER TRANSFORM (DFT) AND FAST FOURIER TRANSFORM (FFT)

Analysis of LTI Discrete Time Systems using DFT, Relation between DTFT and DFT, FFT computations using Decimation in time (DIT) algorithms and Decimation in frequency (DIF) algorithms, Auto correlation, Cross correlation. Realization of recursive and non recursive systems - Direct Form I and Form II - Cascade and parallel realization.

Discrete-time Fourier transform (DTFT)

The Discrete Time Fourier Transform (DTFT) is the member of the Fourier transform family that operates on aperiodic, discrete signals. The best way to understand the DTFT is how it relates to the DFT. To start, imagine that you acquire an N sample signal, and want to find its frequency spectrum. By using the DFT, the signal can be decomposed into sine and cosine waves, with frequencies equally spaced between zero and one-half of the sampling rate. As discussed in the last chapter, padding the time domain signal with zeros makes the period of the time domain longer, as well as making the spacing between samples in the frequency domain narrower. As N approaches infinity, the time domain becomes aperiodic, and the frequency domain becomes a continuous signal. This is the DTFT, the Fourier transform that relates an aperiodic, discrete signal, with a periodic, continuous frequency spectrum.

The mathematics of the DTFT can be understood by starting with the synthesis and analysis equations

$$x[n] = \frac{1}{2\pi}\int_{2\pi} X(\Omega)\, e^{j\Omega n}\, d\Omega \qquad \text{synthesis}$$

$$X(\Omega) = \sum_{n=-\infty}^{+\infty} x[n]\, e^{-j\Omega n} \qquad \text{analysis}$$

$$x[n] \overset{\mathcal{F}}{\longleftrightarrow} X(\Omega)$$

$$X(\Omega) = Re\{X(\Omega)\} + j\, Im\{X(\Omega)\}$$

$$= |X(\Omega)|\, e^{j \measuredangle X(\Omega)}$$

The spectrum of the DTFT is continuous, so either f or ω can be used. The common choice is ω, because it makes the equations shorter by eliminating the always present factor of 2π. Remember, when ω is used, the frequency spectrum extends from 0 to π, which corresponds to DC to one-half of the sampling rate. To make things even more complicated, many authors use Ω (an upper case omega) to represent this frequency in the DTFT, rather than ω (a lower case omega.

PROPERTIES OF THE FOURIER TRANSFORM

$$x[n] \overset{\mathcal{F}}{\longleftrightarrow} X(\Omega)$$

Periodic:

$$X(\Omega) = X(\Omega + 2\pi m)$$

Symmetry:

$$x[n] \text{ real} \Rightarrow X(-\Omega) = X^*(\Omega)$$

$$\left.\begin{matrix} Re\{X(\Omega)\} \\ |X(\Omega)| \end{matrix}\right\} \text{ even}$$

$$\left.\begin{matrix} Im\{X(\Omega)\} \\ \measuredangle X(\Omega) \end{matrix}\right\} \text{ odd}$$

Time shifting:

$$x[n-n_o] \overset{\mathcal{F}}{\longleftrightarrow} e^{-j\Omega n_o} X(\Omega)$$

Frequency shifting:

$$e^{j\Omega_o n} x[n] \overset{\mathcal{F}}{\longleftrightarrow} X(\Omega - \Omega_o)$$

Linearity:

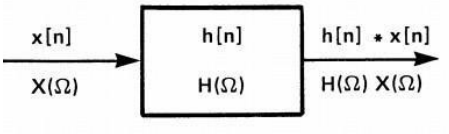
$$ax_1[n] + bx_2[n] \overset{\mathcal{F}}{\longleftrightarrow} aX_1(\Omega) + bX_2(\Omega)$$

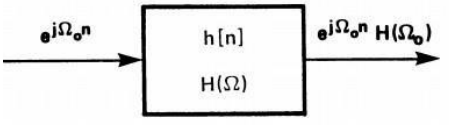
Parseval's relation:

$$\sum_{n=-\infty}^{+\infty} |x[n]|^2 = \frac{1}{2\pi} \int_{2\pi} |X(\Omega)|^2 \, d\Omega$$

CONVOLUTION PROPERTY

$$h[n] * x[n] \overset{\mathcal{F}}{\longleftrightarrow} H(\Omega)\, X(\Omega)$$





Discrete Fourier Transform (DFT):

Definition (**Discrete Fourier Transform**): *Given a finite sequence*

$$x = [x(0), x(1),..., x(N-1)]$$

its Discrete Fourier Transform (DFT) is a finite sequence

$$X = DFT(x) = [X(0), X(1),..., X(N-1)]$$

where

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

Inverse Discrete Fourier Transform (IDFT):

The inverse discrete Fourier transform of X(k) is defined as

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

For notation purpose discrete Fourier transform and inverse Fourier transform can be represented by

$$X(k) = DFT\,[x(n)]$$
$$x(n) = IDFT\,[X(k)]$$

Formula:

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

Where K and n are in the range of 0 ,1,2……N-1 For example, if N=4, K= 0,1,2,3:

N=0,1,2,3 Alternative Formula:

$$X(k) = \sum_{n=0}^{N-1} x(n)W^{kn} \quad \longleftarrow W = e^{-j\frac{2\pi}{N}}$$

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

Properties of DFT:

Periodicity property:

If X(k) is the N-point DFT of x(n), then

$$X(k+N)=X(k)$$

Linearity property:

If $X_1(k)=DFT[x_1(n)]$ & $X_2(k)=DFT[x_2(n)]$, then

$$DFT[a_1x_1(n)+a_2x_2(n)]=a_1X_1(k)+a_2X_2(k)$$

Convolution property:

If $X_1(k) = DFT[x_1(n)]$ & $X_2(k) = DFT[x_2(n)]$, then

$$DFT[x(n) \textcircled{N} x2(n)] = X1(k)X2(k)$$

Where $\textcircled{N}$ indicates N-point circular convolution.

Multiplication property:

If $X_1(k) = DFT[x_1(n)]$ & $X_2(k) = DFT[x_2(n)]$, then

$$DFT[x1(n)x2(n)] = (1/N)[X1(k) \textcircled{N} X2(k)]$$

Where $\textcircled{N}$ Indicates N-point circular convolution.

Time reversal property:

If X(k) is the N-point DFT of x(n), then DFT[x(N□n)] = X(N□k)

Time shift property:

If X(k) is the N-point DFT of x(n), then

$$DFT\left\{x((n-m))_N\right\} = X(k)\, e^{-j\frac{2\pi km}{N}}$$

Symmetry properties:

If x(n)=x_R(n)+jx_I(n) is N-point complex sequence and X(k)=X_R(k)+jX_I(k) is the N- point DFT of x(n) where x_R(n) & x_I(n) are the real & imaginary parts of x(n) and X_R(k) & X_I(k) are the those of X(k), then

(i) **DFT[x^*(n)]=X^*(N −k)**

(ii) **DFT[x^*(N −n)]=X^*(k)**

(iii) **DFT[x_R(n)]=(1/2)[X(k)+X^*(N−k)]**

(iv) **DFT[x_I(n)]=(1/2j)[X(k)−X^*(N−k)]**

(v) **DFT[x_{ce}(n)]=X_R(k) where x_{ce}(n)=(1/2)[x(n)+x^*(N−n)]**

(vi) **DFT[x_{co}(n)]=jX_I(k) where x_{co}(n)=(1/2)[x(n)−x^*(N−n)]**

If x(n) is real, then

(i) **If x(n) is real, then**

a. **X(k)=X^*(N−k)**

b. **X_R(k)=X_R(N−k)**

(ii) **If x(n) is real, then**

a) **X(k)=X^*(N-k)**

b) **X_R(k)=X_R(N-k)**

c) **X_I(k)= -X_I(N-k)**

d) **|X(k)|=|X(N-k)|**

e) **|X(k)|=|X(N-k)|**

f) **ARG X(k)= ANG X(N-k)**

(i) **DFT[x_{ce}(n)]=X_R(k) where x_{ce}(n)=(1/2)[x(n)+x(N+n)]**

(ii) **DFT[x_{co}(n)]=jX_I(k) where x_{co}(n)=(1/2)[x(n)-x(N-n)]**

Problem

Compute 4-point DFT and 8-point DFT of causal three sample sequence given by

$$x(n) = \frac{1}{3} \;;\; 0 \le n \le 2$$
$$= 0 \;;\; \text{else}$$

Solution

By the definition of N-point DFT, the k^{th} complex coefficient of X(k), for $0 \le k \le N-1$, is given by,

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

a) 4-point DFT (\ N = 4)

$$X(k) = \sum_{n=0}^{4-1} x(n)\, e^{\frac{-j2\pi kn}{4}} = \sum_{n=0}^{2} x(n)\, e^{\frac{-j\pi kn}{2}} = x(0)\, e^{0} + x(1)\, e^{\frac{-j\pi k}{2}} + x(2)\, e^{-j\pi k}$$

$e^{\pm j\theta} = \cos\theta \pm j\sin\theta$

$$= \frac{1}{3} + \frac{1}{3} e^{\frac{-j\pi k}{2}} + \frac{1}{3} e^{-j\pi k} = \frac{1}{3}\left[1 + \cos\frac{\pi k}{2} - j\sin\frac{\pi k}{2} + \cos\pi k - j\sin\pi k\right]$$

For 4-point DFT, X(k) has to be evaluated for k = 0, 1, 2, 3.

When $k = 0$; $X(0) = \frac{1}{3}[1 + \cos 0 - j\sin 0 + \cos 0 - j\sin 0]$

$$= \frac{1}{3}(1 + 1 - j0 + 1 - j0) = 1 = 1\angle 0$$

When $k = 1$; $X(1) = \frac{1}{3}\left[1 + \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} + \cos\pi - j\sin\pi\right]$

$$= \frac{1}{3}(1 + 0 - j - 1 - j0) = -j\frac{1}{3} = \frac{1}{3}\angle -\pi/2 = 0.333\angle -0.5\pi$$

When $k = 2$; $X(2) = \frac{1}{3}[1 + \cos\pi - j\sin\pi + \cos 2\pi - j\sin 2\pi]$

$$= \frac{1}{3}(1 - 1 - j0 + 1 - j0) = \frac{1}{3} = 0.333\angle 0$$

When $k = 3$; $X(3) = \frac{1}{3}\left[1 + \cos\frac{3\pi}{2} - j\sin\frac{3\pi}{2} + \cos 3\pi - j\sin 3\pi\right]$

$$= \frac{1}{3}(1 + 0 + j - 1 - j0) = j\frac{1}{3} = \frac{1}{3}\angle\pi/2 = 0.333\angle 0.5\pi$$

\ The 4-point DFT sequence X(k) is given by,

$$X(k) = \{\, 1\angle 0,\quad 0.333\angle -0.5\pi,\quad 0.333\angle 0,\quad 0.333\angle 0.5\pi \,\}$$

$\therefore$ Magnitude Function, $|X(k)| = \{\, 1,\ 0.333,\ 0.333,\ 0.333 \,\}$

Phase Function, $\angle X(k) = \{0,\ -0.5\pi,\ 0,\ 0.5\pi\}$

Phase angles are in radians.

b) 8-point DFT (\ N = 8)

$$X(k) = \sum_{n=0}^{8-1} x(n)\, e^{\frac{-j2\pi kn}{8}} = \sum_{n=0}^{2} x(n)\, e^{\frac{-j\pi kn}{4}} = x(0)\, e^{0} + x(1)\, e^{\frac{-j\pi k}{4}} + x(2)\, e^{\frac{-j\pi k}{2}}$$

$e^{\pm j\theta} = \cos\theta \pm j\sin\theta$

$$= \frac{1}{3} + \frac{1}{3} e^{\frac{-j\pi k}{4}} + \frac{1}{3} e^{\frac{-j\pi k}{2}} = \frac{1}{3}\left[1 + \cos\frac{\pi k}{4} - j\sin\frac{\pi k}{4} + \cos\frac{\pi k}{2} - j\sin\frac{\pi k}{2}\right]$$

For 8-point DFT, X(k) has to be evaluated for k = 0, 1, 2, 3, 4, 5, 6, 7.

When k = 0 ; $X(0) = \frac{1}{3}[1 + \cos 0 - j\sin 0 + \cos 0 - j\sin 0]$

$= \frac{1}{3}(1 + 1 - j0 + 1 - j0) = 1 = 1\angle 0$

When k = 1 ; $X(1) = \frac{1}{3}\left[1 + \cos\frac{\pi}{4} - j\sin\frac{\pi}{4} + \cos\frac{\pi}{2} - j\sin\frac{\pi}{2}\right]$

$= 0.333\,(1 + 0.707 - j0.707 + 0 - j1)$

$= 0.568 - j0.568 = 0.803\angle -0.785 = 0.803\angle -0.25\pi$

$\frac{0.785}{\pi} \times \pi = 0.25\pi$

When k = 2 ; $X(2) = \frac{1}{3}\left[1 + \cos\frac{2\pi}{4} - j\sin\frac{2\pi}{4} + \cos\frac{2\pi}{2} - j\sin\frac{2\pi}{2}\right]$

$= 0.333\,(1 + 0 - j1 - 1 - j0)$

$= -j0.333 = 0.333\angle -\pi/2 = 0.333\angle -0.5\pi$

When k = 3 ; $X(3) = \frac{1}{3}\left[1 + \cos\frac{3\pi}{4} - j\sin\frac{3\pi}{4} + \cos\frac{3\pi}{2} - j\sin\frac{3\pi}{2}\right]$

$= 0.333\,(1 - 0.707 - j0.707 + 0 + j1)$

$= 0.098 + j0.098 = 0.139\angle 0.785 = 0.139\angle 0.25\pi$

When k = 4 ; $X(4) = \frac{1}{3}\left[1 + \cos\frac{4\pi}{4} - j\sin\frac{4\pi}{4} + \cos\frac{4\pi}{2} - j\sin\frac{4\pi}{2}\right]$

$= 0.333\,(1 - 1 - j0 + 1 - j0) = 0.333 = 0.333\angle 0$

When k = 5 ; $X(5) = \frac{1}{3}\left[1 + \cos\frac{5\pi}{4} - j\sin\frac{5\pi}{4} + \cos\frac{5\pi}{2} - j\sin\frac{5\pi}{2}\right]$

$= 0.333\,(1 - 0.707 + j0.707 + 0 - j1)$

$= 0.098 - j0.098 = 0.139\angle -0.785 = 0.139\angle -0.25\pi$

When k = 6 ; $X(6) = \frac{1}{3}\left[1 + \cos\frac{6\pi}{4} - j\sin\frac{6\pi}{4} + \cos\frac{6\pi}{2} - j\sin\frac{6\pi}{2}\right]$

$= 0.333\,(1 + 0 + j1 - 1 - j0)$

$= j0.333 = 0.333\angle \pi/2 = 0.333\angle 0.5\pi$

When k = 7 ; $X(7) = \frac{1}{3}\left[1 + \cos\frac{7\pi}{4} - j\sin\frac{7\pi}{4} + \cos\frac{7\pi}{2} - j\sin\frac{7\pi}{2}\right]$

$= 0.333\,(1 + 0.707 + j0.707 + 0 + j1)$

$= 0.568 + j0.568 = 0.803\angle 0.785 = 0.803\angle 0.25\pi$

Phase angles are in radians.

\ The 8-point DFT sequence X(k) is given by,

$X(k) = \{1\angle 0,\ 0.803\angle -0.25\pi,\ 0.333\angle -0.5\pi,\ 0.139\angle 0.25\pi,\ 0.333\angle 0,\ 0.139\angle -0.25\pi,$

$0.333\angle 0.5\pi,\ 0.803\angle 0.25\pi\}$

$\therefore$ Magnitude Function, $|X(k)| = \{1,\ 0.803,\ 0.333,\ 0.139,\ 0.333,\ 0.139,\ 0.333,\ 0.803\}$

Phase Function, $\angle X(k) = \{0,\ -0.25\pi,\ -0.5\pi,\ 0.25\pi,\ 0,\ -0.25\pi,\ 0.5\pi,\ 0.25\pi\}$

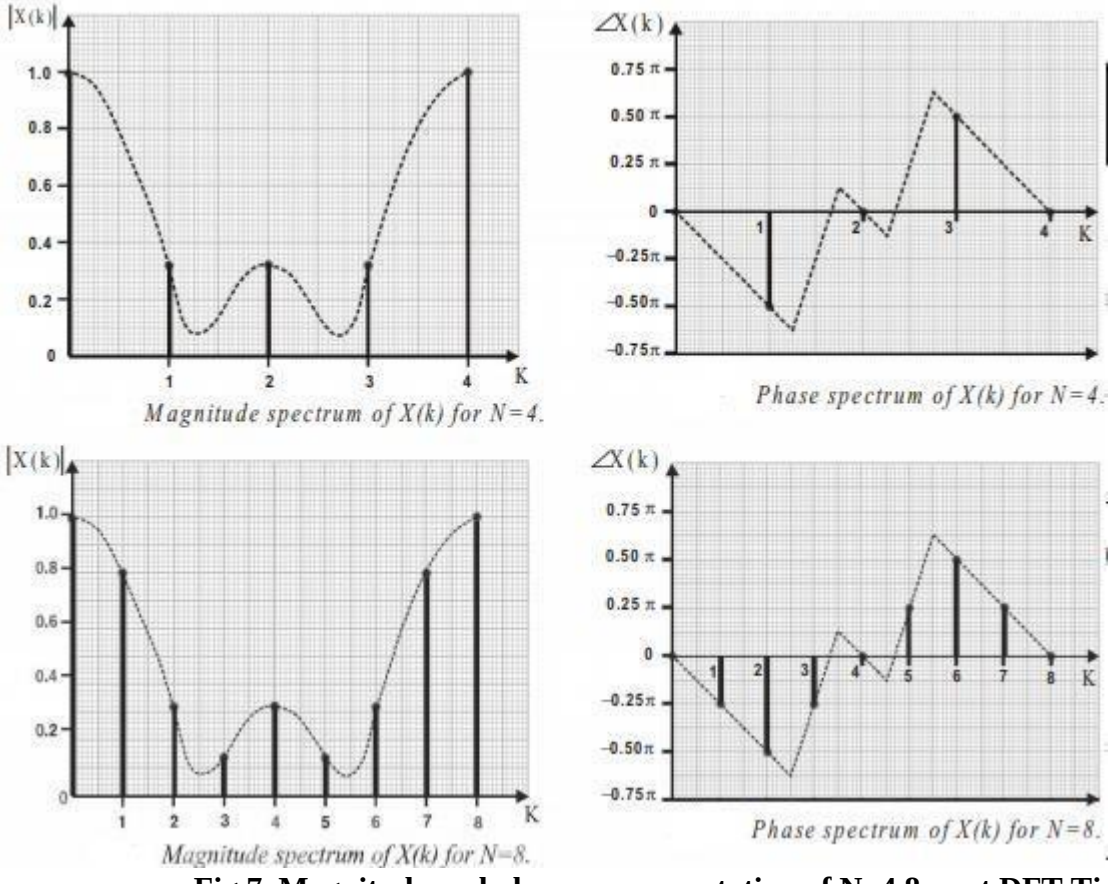


Fig 7. Magnitude and phasor representation of N=4,8 pont DFT Time Signals

[courtesy: DSP by Nagoorkani]

Fast Fourier Transform (FFT)

The Fast Fourier Transform (FFT) is a method (or algorithm) for computing the discrete Fourier transform (DFT) with reduced number of calculations. The computational efficiency is achieved if we adopt a divide and conquer approach. This approach is based on the decomposition of an N-point DFT into successively smaller DFTs. This basic approach leads to a family of an efficient computational algorithms known collectively as FFT algorithms. Radix-r FFT In an N-point sequence, if N can be expressed as $N = r^m$, then the sequence can be decimated into r-point sequences. For each r-point sequence, r-point DFT can be computed. From the results of r-point DFT, the r2 -point DFTs are computed. From the results of r2 -point DFTs, the r3 -point DFTs are computed and so on, until we get r^m point DFT. This FFT algorithm is called radix-r FFT. In computing N-point DFT by this method the number of stages of computation will be m times.

Radix-2 FFT For radix-2 FFT, the value of N should be such that, $N = 2^m$, so that the N-point sequence is decimated into 2-point sequences and the 2-point DFT for each decimated

sequence is computed. From the results of 2-point DFTs, the 4-point DFTs can be computed. From the results of 4-point DFTs, the 8-point DFTs can be computed and so on, until we get N-point DFT.

Number of Calculations in N-point DFT

N^2 number of complex multiplications and N(N – 1) number of complex additions

Number of Calculations in Radix-2 FFT

$N/2\log_2 N$ complex multiplications and $N \log_2 N$ complex additions.

Radix-2 FFT algorithms:

Decimation-In-Time (DIT) FFT algorithm:

The algorithm in which the decimation is based on splitting the sequence x(n) into successively smaller sequences is called the decimation-in-time algorithm.

The N-point DFT of a sequence x(n) is given by

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

where $W_N = e^{-j(2\pi/N)}$. X(k) is periodic with period N i.e., X(k+N)=X(k).

Splitting Equ(1) into two, one for even-indexed samples of x(n) and the other for odd- indexed samples of x(n), we have

$$X(k) = \sum_{n\ even} x(n)W_N^{nk} + \sum_{n\ odd} x(n)W_N^{nk} \qquad (2)$$

Substituting n=2n for n even and n=2n+1 for n odd, we have

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

8-Point DFT Using Radix-2 DIT FFT

The input sequence is 8-point sequence. Therefore, $N = 8 = 2^3 = r^m$. Here, r = 2 and m = 3. Therefore, the computation of 8-point DFT using radix-2 FFT, involves three stages of computation. The given 8-point sequence is decimated to 2-point sequences. For each 2-point sequence, the 2-point DFT is computed. From the results of 2-point DFT, the 4-point DFT can be computed. From the results of 4-point DFT, the 8-point DFT can be computed.

Let the given sequence be x(0), x(1), x(2), x(3), x(4),x(5), x(6), x(7), which consists of 8 samples. The 8-samples should be decimated into sequences of 2-samples. Before decimation they are arranged in bit reversed order, as shown in table

Normal order		Bit reversed order	
x(0)	x(000)	x(0)	x(000)
x(1)	x(001)	x(4)	x(100)
x(2)	x(010)	x(2)	x(010)
x(3)	x(011)	x(6)	x(110)
x(4)	x(100)	x(1)	x(001)
x(5)	x(101)	x(5)	x(101)
x(6)	x(110)	x(3)	x(011)
x(7)	x(111)	x(7)	x(111)

Fig 8. Bit reversal order of 2 point DFT

Using the decimated sequences as input the 8-point DFT is computed. The fig shows the three stages of computation of an 8-point DFT.

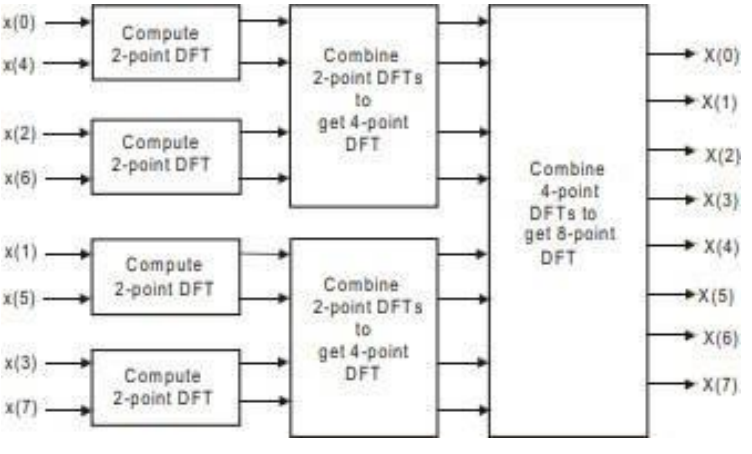


Fig 9.Block diagram representation of 8 pt DFT

Flow Graph for 8-Point DFT using Radix-2 DIT FFT

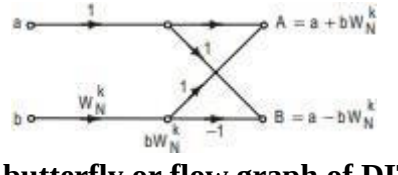


Fig 10.Basic butterfly or flow graph of DIT rad ix-2 FFT.

The signal flow graph is also called butterfly diagram since it resembles a butterfly

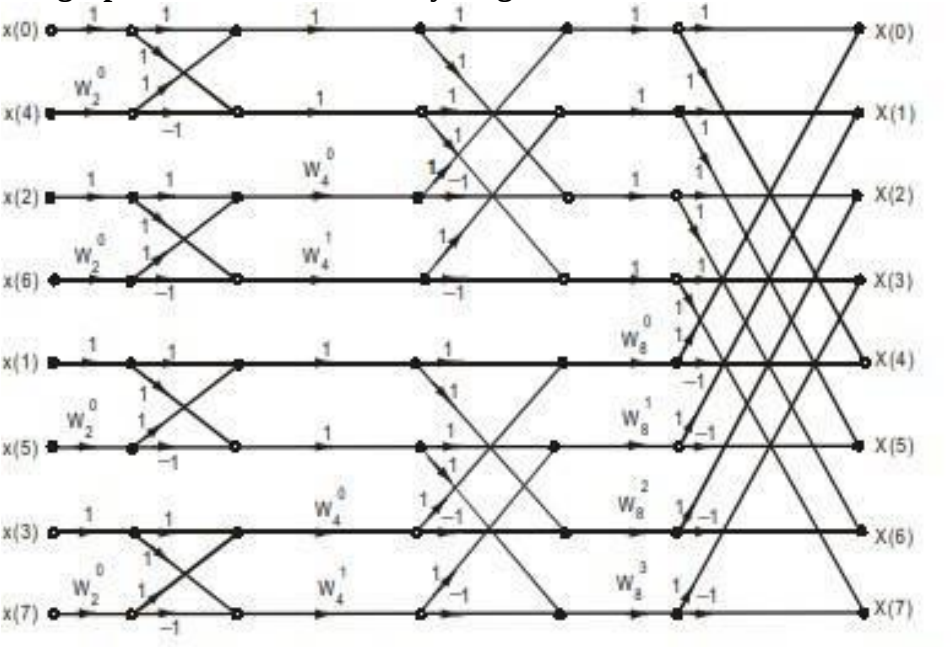


Fig 11. *The flow graph (or butterfly diagram) for 8-point DFT via radix-2 DIT FFT.*

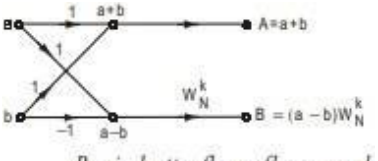
8-point DFT Using Radix-2 DIF FFT

The DIF computation for an eight sequence is discussed in detail in this section. Let x(n) be an 8-point sequence. Therefore N = 8 = 2^3 = r^m. Here, r = 2 and m = 3. Therefore, the computation of 8-point DFT using radix-2 FFT involves three stages of computation. The samples of x(n) are,

x(0), x(1), x(2), x(3), x(4), x(5), x(6), x(7).

Flow Graph For 8-point DFT using Radix-2 DIF FFT

The above basic computation can be expressed by a signal flow graph shown in Fig



Basic butterfly or flow graph of DIF radix-2 FFT.

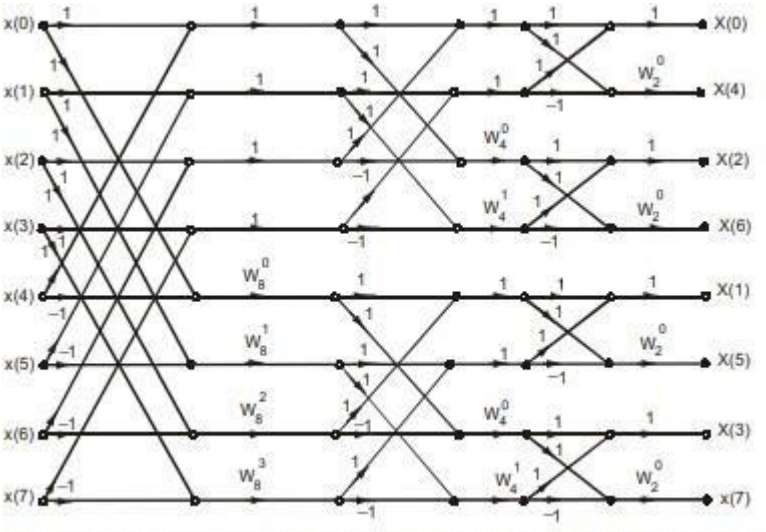


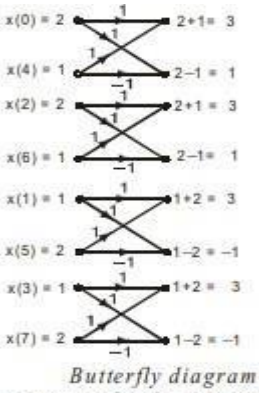
Fig 12. *The flow graph (or butterfly diagram) for 8-point DFT via radix-2 DIF FFT.*

Problem:
An 8-point sequence is given by x(n) = {2, 1, 2, 1, 1, 2, 1, 2}. Compute 8-point DFT of x(n) by a) radix-2 DIT-FFT and b) radix-2 DIF-FFT. Also sketch the magnitude and phase spectrum.

a) <u>8-point DFT by Radix-2 DIT-FFT</u>

The given sequence is first arranged in the bit reversed order

The sequence x(n) in normal order	The sequence x(n) in bit reversed order
x(0) = 2	x(0) = 2
x(1) = 1	x(4) = 1
x(2) = 2	x(2) = 2
x(3) = 1	x(6) = 1
x(4) = 1	x(1) = 1
x(5) = 2	x(5) = 2
x(6) = 1	x(3) = 1
x(7) = 2	x(7) = 2



Butterfly diagram for first stage of radix-2 DIT FFT.

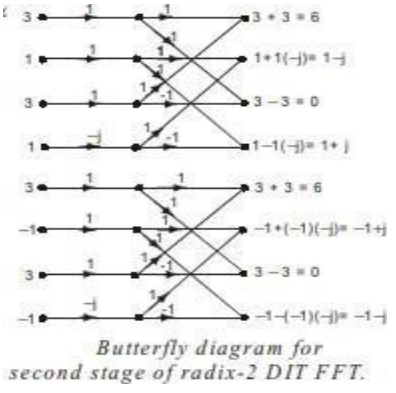
The 8-point DFT by radix-2 FFT involve 3 stages of computation with 4-butterfly computations in each stage. The sequence rearranged in the bit reversed order forms the input to the first stage. For other stages of computation the output of previous stage will be the input for current stage.

<u>Second stage computation</u>

The input sequence to second stage computation = { 3, 1, 3, 1, 3, 1, 3, 1 }
The phase factors involved in second stage computation are W_4^0 and W_4^1

$$W_4^0 = e^{-j2\pi \times \frac{0}{4}} = e^0 = 1$$

$$W_4^1 = e^{-j2\pi \times \frac{1}{4}} = e^{-j \times \frac{\pi}{2}} = \cos\left(\frac{-\pi}{2}\right) + j\sin\left(\frac{-\pi}{2}\right) = -j$$



Butterfly diagram for second stage of radix-2 DIT FFT.

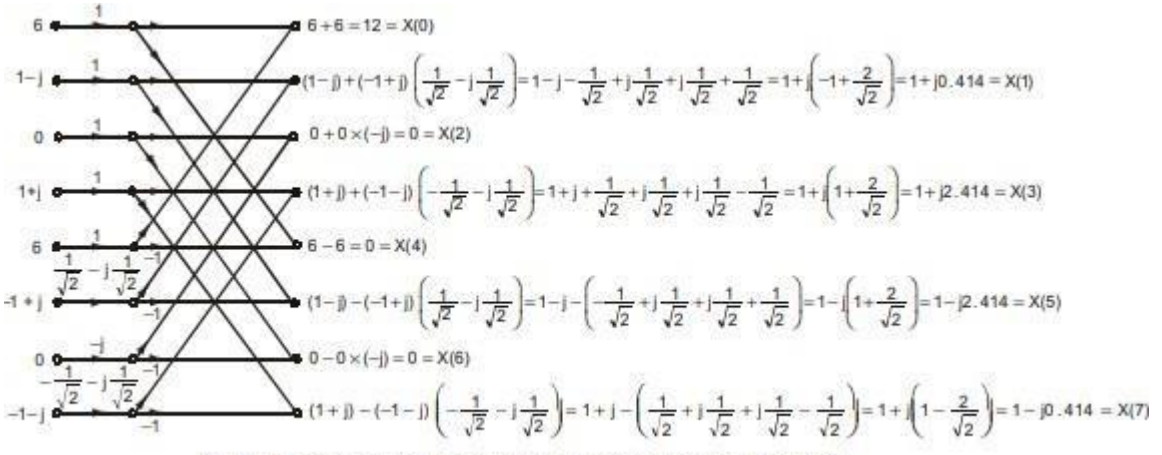
Third stage computation The input sequence to third stage computation = {6, 1j, 0, 1+j, 6, 1+j, 0, 1j} The phase factors involved in third stage computation are W_8^0, W_8^1, W_8^2 and W_8^3

$$W_8^0 = e^{-j2\pi \times \frac{0}{8}} = e^0 = 1$$

$$W_8^1 = e^{-j2\pi \times \frac{1}{8}} = e^{-j \times \frac{\pi}{4}} = \cos\left(\frac{-\pi}{4}\right) + j\sin\left(\frac{-\pi}{4}\right) = \frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$$

$$W_8^2 = e^{-j2\pi \times \frac{2}{8}} = e^{-j \times \frac{\pi}{2}} = \cos\left(\frac{-\pi}{2}\right) + j\sin\left(\frac{-\pi}{2}\right) = -j$$

$$W_8^3 = e^{-j2\pi \times \frac{3}{8}} = e^{-j \times \frac{3\pi}{4}} = \cos\left(\frac{-3\pi}{4}\right) + j\sin\left(\frac{-3\pi}{4}\right) = -\frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$$



Butterfly diagram for third stage of radix-2 DIT FFT of X(k) .

Fig 13. Butterfly diagram for third stage of radix-2 DIT FFT

b) 8-point DFT by Radix-2 DIF-FFT

For 8-point DFT by radix-2 FFT we require 3-stages of computation with 4-butterfly computation in each stage. The given sequence is the input to first stage. For other stages of computations, the output of previous stage will be the input for current stage.

First stage computation

The input sequence for first stage of computation = { 2, 1, 2, 1, 1, 2, 1, 2 }

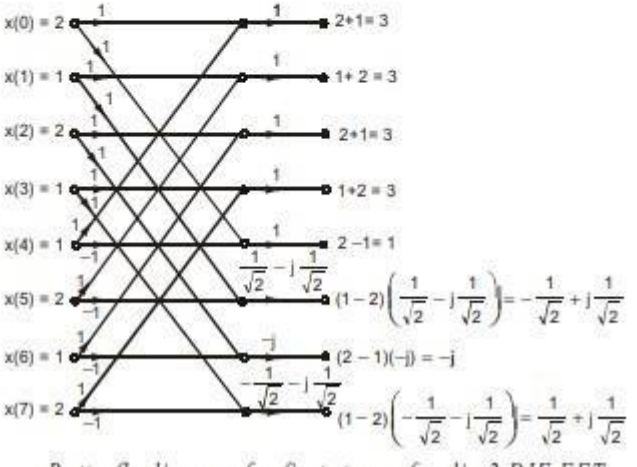
The phase factors involved in first stage computation are W_8^0 , W_8^1 , W_8^2 and W_8^3

$$W_8^0 = e^{-j2\pi \times \frac{0}{8}} = 1$$

$$W_8^1 = e^{-j2\pi \times \frac{1}{8}} = e^{-j\frac{\pi}{4}} = \cos\left(-\frac{\pi}{4}\right) + j\sin\left(-\frac{\pi}{4}\right) = \frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$$

$$W_8^2 = e^{-j2\pi \times \frac{2}{8}} = e^{-j\frac{\pi}{2}} = \cos\left(-\frac{\pi}{2}\right) + j\sin\left(-\frac{\pi}{2}\right) = -j$$

$$W_8^3 = e^{-j2\pi \times \frac{3}{8}} = e^{-j\frac{3\pi}{4}} = \cos\left(-\frac{3\pi}{4}\right) + j\sin\left(-\frac{3\pi}{4}\right) = -\frac{1}{\sqrt{2}} - j\frac{1}{\sqrt{2}}$$



Butterfly diagram for first stage of radix-2 DIF FFT.

Fig 14. Butterfly diagram for first stage of radix-2 DIT FFT

The output sequence of first stage of computation $= \left\{3,\ 3,\ 3,\ 3,\ 1,\ -\frac{1}{\sqrt{2}} + j\frac{1}{\sqrt{2}},\ -j,\ \frac{1}{\sqrt{2}} + j\frac{1}{\sqrt{2}}\right\}$

Second stage computation

The input sequence for second stage of computation =

$\left\{3,\ 3,\ 3,\ 3,\ 1,\ -\frac{1}{\sqrt{2}} + j\frac{1}{\sqrt{2}},\ -j,\ \frac{1}{\sqrt{2}} + j\frac{1}{\sqrt{2}}\right\}$

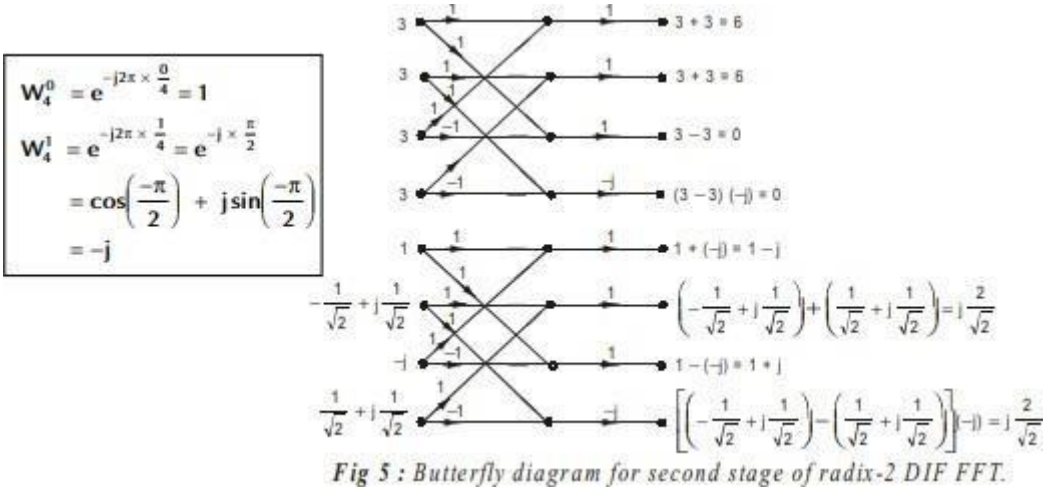
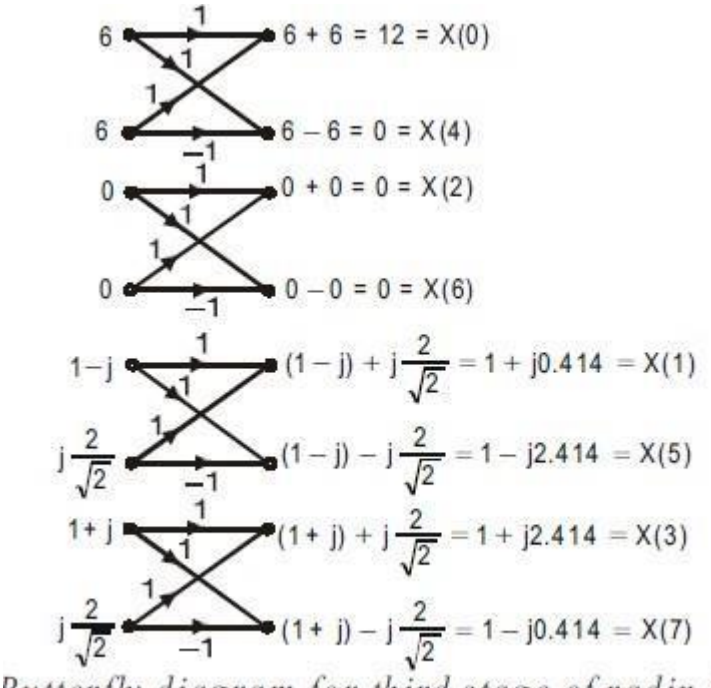


Fig 5 : Butterfly diagram for second stage of radix-2 DIF FFT.

Fig 15. Butterfly diagram for second stage of radix-2 DIT FFT

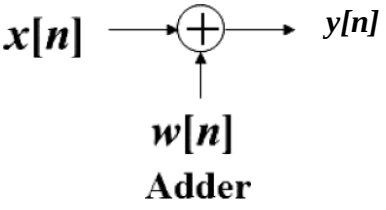
The output sequence of second stage of computation

$= \left\{6,\ 6,\ 0,\ 0,\ 1-j,\ j\frac{2}{\sqrt{2}},\ 1+j,\ j\frac{2}{\sqrt{2}}\right\}$ **Third stage computation**

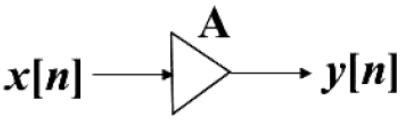


Butterfly diagram for third stage of radix-2 DIF FFT.

Fig 16. Butterfly diagram for third stage of radix-2 DIT FFT

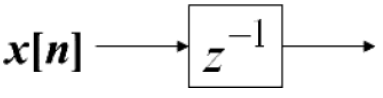


Adder

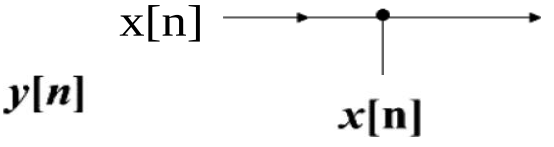


Multiplier

x[n]



Unit delay



A realization is canonic if the realization uses minimum number of delay units. Two realizations are equivalent if they have the same transfer function.

Transpose operation generates an equivalent structure from a given realization by the following steps:

- **Interchange input and output nodes**
- **Reverse all the paths**
- **Replace pick — off nodes with adders and vice — versa.**

FIR Filter Structures

An FIR filter has a system function given by

$$H(z) = \sum_{k}^{M} b_k z^{-k} = b_0 + b_1 z^{-1} + \cdots\cdots + b_M z^{-M}$$

From this equation, the impulse response h[n] can be written as

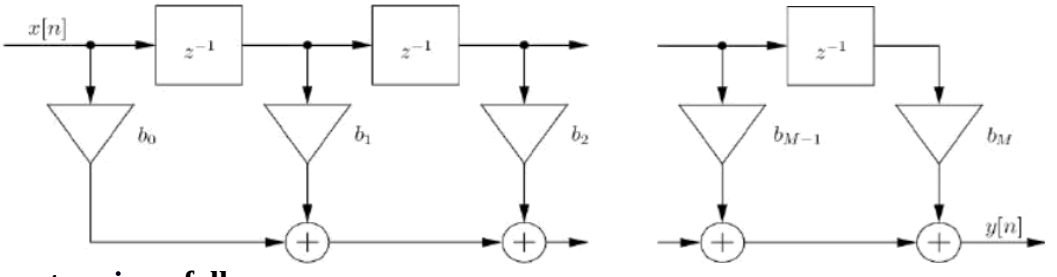
$$h[n] = \begin{matrix} bj \ , 0 \ \tilde{n} \ n \ DIY \\ O \qquad otherwise \end{matrix}$$

the difference equation representation is given by

$$y[n] = b\ x[n] + box\ [n - 1\ ] - F\ box[n - 2] - F\text{--- -- - -} F\ by{:}x\ [n - M\ ]$$

The length of the filter is M+1, and the oriler of the filter is M.

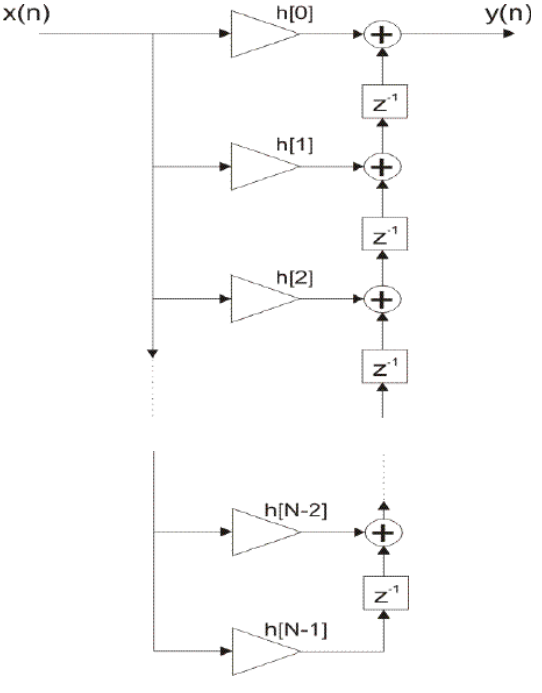
1. **Direct — Form/Transversal/Tapped Delav Line Structure From the above equation, the direct form**



structure is as follows

Here, the filter has an order of M, it requires M delays/memory locations, M+1 Multipliers, and M adders.
Structures in which the multiplier coefficients are directly available as coefficients of H(z) are called Direct — Form Structures.

2. **Transposeil version of direct form**
By applying the steps to obtain the transposed form to the above direct — form structure, we obtain the following transposed sdvcture.



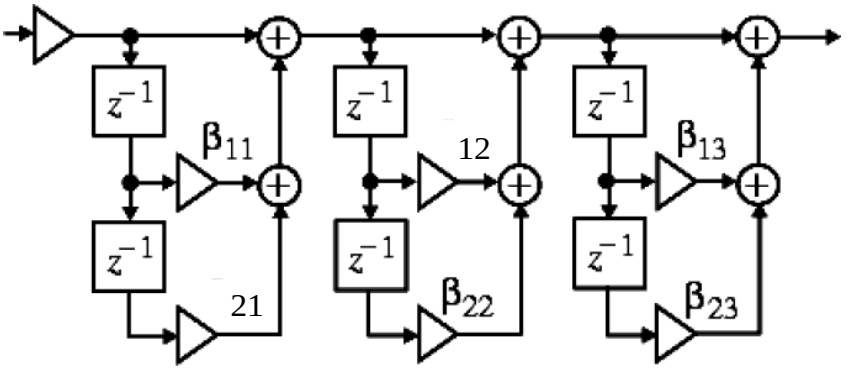
3. **Cascade form structure**
To obtain the cascade sdvcture, H(z) is factorized in terms of second — order factors and lust — order factors.

$$H(z) = by - F\ b \bullet \ ^{1} + - - + bp \quad z \ (\text{“} \ ^{1}$$

$$H(z) = \begin{cases} b_0 \left\{ \prod_{k=1}^{(M-1)/2} (1 + B_{1k}z^{-1} + B_{2k}z^{-2}) \right\} & \textit{for M odd} \\ b_0 \left\{ (1 + b_{10}z^{-1}) \prod_{k=1}^{(M-1)/2} (1 + B_{1k}z^{-1} + B_{2k}z^{-2}) \right\} & \textit{for M even} \end{cases}$$

For example, for M=7 (order =6), the cascade sdvcture would be

fi[0]



Where h[0]=bo

Basic Structures for HR Systems

The convolution sum description of an LTI discrete — time system can, in principle, be used to implement the system. However, for an HR system, this approach is not practical, since the impulse response is infinite in length. So, we use the input — output relation to obtain the realization.

The system function of an HR filter is

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

Order = N-1

The corresponding difference equation is

$$\textbf{y[n]} = - \sum_{k=1}^{N-1} \text{any[n} - k] \text{ -1- } \sum_{k=0}^{M-1} \textit{box [n} - \textit{k]}$$

Direct Form — I Structure

The transfer function H(z) of the HR system is divided into two parts connected in cascade, with the first part Hi(z) containing only the zeroes, and the second part H2(z) containing only the

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

poles.

Where

$$H_1(z) = \sum_{k=0} b_k z^{-k}$$

And

"(')' $\frac{1}{+\sum_{k=1}^{N-1} a_k z^{-k}}$

These equations can be rewritten as

$$H_1(z) = \frac{\mathbf{W(z)}}{X(z)} = \sum_{k=0} \boldsymbol{bk} \quad '$$

or in time domain

w[n] = *b* x[n] + *box [n* — 1] -F- - -- - + bye [n — M]

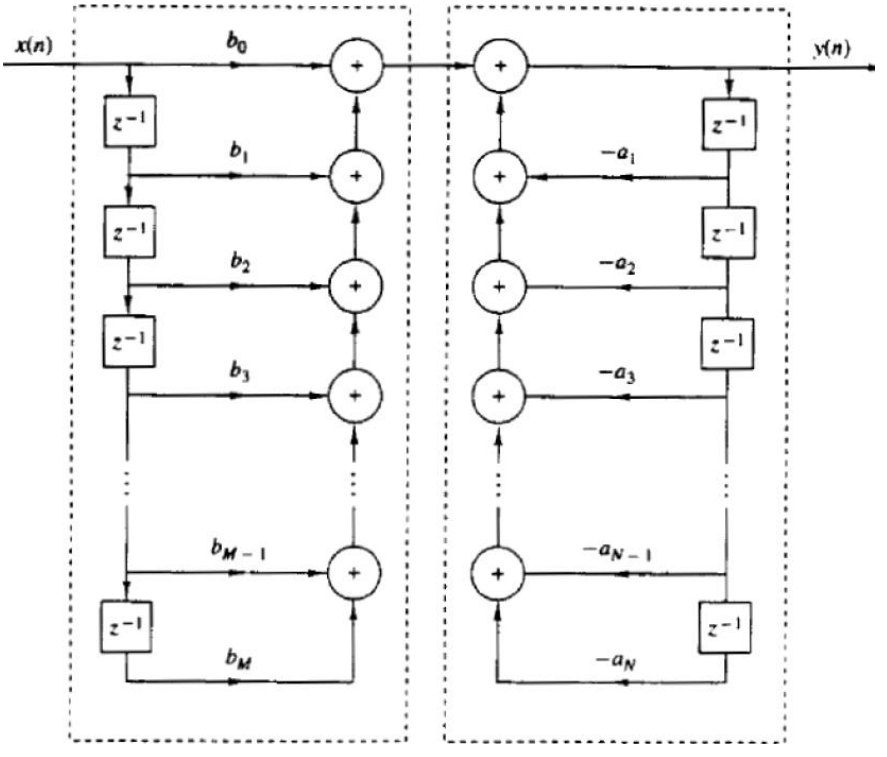
And

$$H_2(z) = \frac{\boldsymbol{Y\{z\ )}}{W(z)} \quad \frac{1}{1 + \sum_{k=1}^{N} a_k z^{-k}}$$

Or, in time domain,

y[n] = w[n] — any[n — 1] — my [n — 2] — $a_N y[n - N]$

Realizing the above two equations for Hi(z) and H2(z) using basic building blocks and connecting them in cascade, we obtain the Direct Form — I structure as follows



This realization requires M+N memory units, M+N+1 multipliers and M+N adders.

Direct Form — II Structure

Since, in a cascade aiTangement, the order of the systems is not important, the all — pole system H2(z) and the all - zero system Hi(z) can be interchanged .i.e.,

$$H_1(z) = \frac{}{X(z)} - \frac{}{1 + \sum_{k=1}^{N} a_k z^{-k}}$$

K(z) _ 1

Or, in time domain,

r[n] = x[n] — rim[n — 1] — rim[n — 2] — *N /• — N]*

And

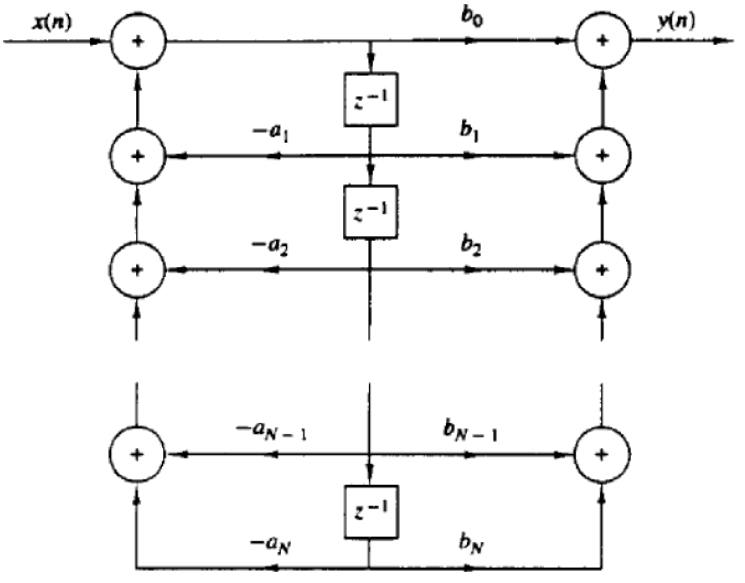
$$H_1(z) = \frac{Y(z)}{V(z)} = \sum_{k=0} b_k z^{-k}$$

or in lime domain

y [n] = *bqv [n]* -1- bar[n — 1] -F *by v [n — M]*

-I-- - - -

Both the time domain equations involve the delayed versions of v[n], and hence require a single set of delay elements. This results in the Direct — Form II structure as follows



This structure requires M+N+1 multipliers, M+N additions and maximum(M, N} delays. Since this structure minimizes the number of delays, it is called canonic.

Cascade — Form Structures

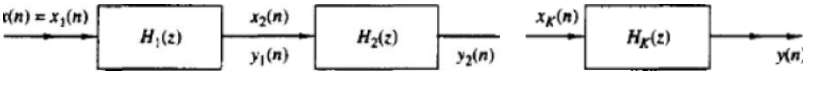
The system function of an HR filter is given by

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

The system is factored into a cascade of second — order subsystems, such that H(z) can be expressed as

$$H\{z\) — \prod_{r=l} H;\{z\), K — tritegral\ part\ of\ \frac{N+1}{2}.$$

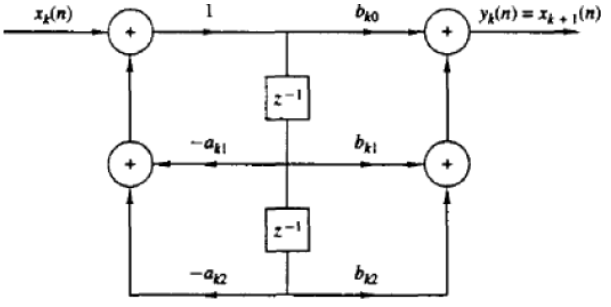
IfN is odd, one of the subsystems is of first order.



In the above equation, each of the subsystems H,(z) has the general form

$$H_i(z) = \frac{b_{i0} + b_{i1} z^{-1} + b_{i2} z^{-2}}{1 + a_{i1} z^{-1} + a_{i2} z^{-2}}.$$

Each of the second — order subsystem can be realized either in Direct Form — I, or Direct — Form II or transposed form.



Since there are many ways of pairing poles and zeroes, a variety of cascade realizations are possible.

Parallel — Form Structures

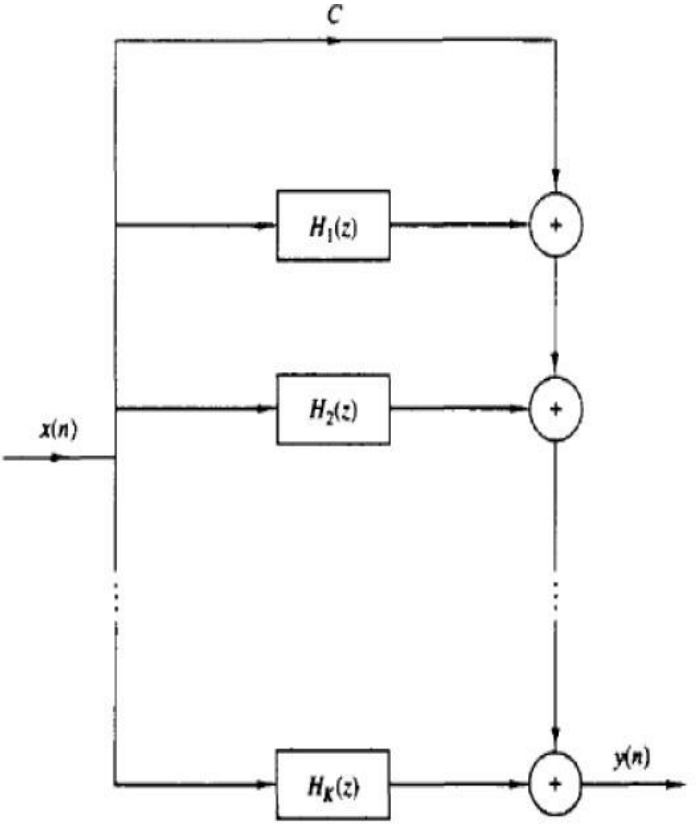
A parallel form realization of an HR system can be obtained by performing partial — fraction expansion of
H(z) .i.e.,

$$H(z) = C + \sum_{k=1} \frac{A_k}{1 - p_k z^{-1}}.$$

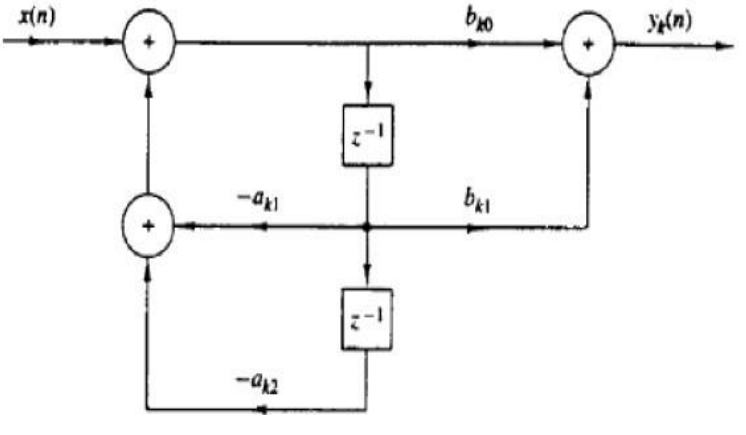
If some poles are complex valued, pairs of complex conjugate poles are combined to get second order subsystems.

H(z) *C* -F *p* _- ffg(z), where

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



ıch of the subsystems can be realized either in Direct Form – I, or Direct – Form II or transposed foı



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF ELECTRICAL AND ELECTRONICS

DEPARTMENT OF ELECTRONICS AND COMMMUNICATION ENGINEERING

UNIT - III
Digital Filter Design

UNIT 3 DIGITAL FILTER DESIGN

Design of IIR filters using Impulse invariant and Bilinear transformation method. Review of Butterworth and Chebyshev approximations, Frequency selective filters: Ideal filter characteristics, low pass, high pass and band pass filters -Linear Phase FIR filter–Phase delay– Group delay– Design of FIR filter using window method - Rectangular, Hanning and Hamming Windows.

Symmetric and Antisymmetric FIR filters

FIR filters are digital filters with finite impulse response. They are also known as on- recursive digital filters as they do not have the feedback (a recursive part of a filter), even though recursive algorithms can be used for FIR filter realization. FIR filters can be designed using different methods, but most of them are based on ideal filter approximation. The objective is not to achieve ideal characteristics, as it is impossible anyway, but to achieve sufficiently good characteristics of a filter. The transfer function of FIR filter approaches the ideal as the filter order increases, thus increasing the complexity and amount of time needed for processing input samples of a signal being filtered. The resulting frequency response can be a monotone function or an oscillatory function within a certain frequency range. The waveform of frequency response depends on the method used in design process as well as on its parameters.

This chapter describes the most popular method for FIR filter design that uses window functions. The characteristics of the transfer function as well as its deviation from the ideal frequency response depend on the filter order and window function in use.

Each filter category has both advantages and disadvantages. This is the reason why it is so important to carefully choose category and type of a filter during design process.

Obviously, in such cases when it is necessary to have a linear phase characteristic, FIR filters are the only option available. If the linear phase characteristic is not necessary, as is the case with processing speech signals, FIR filters are not good solution at all.

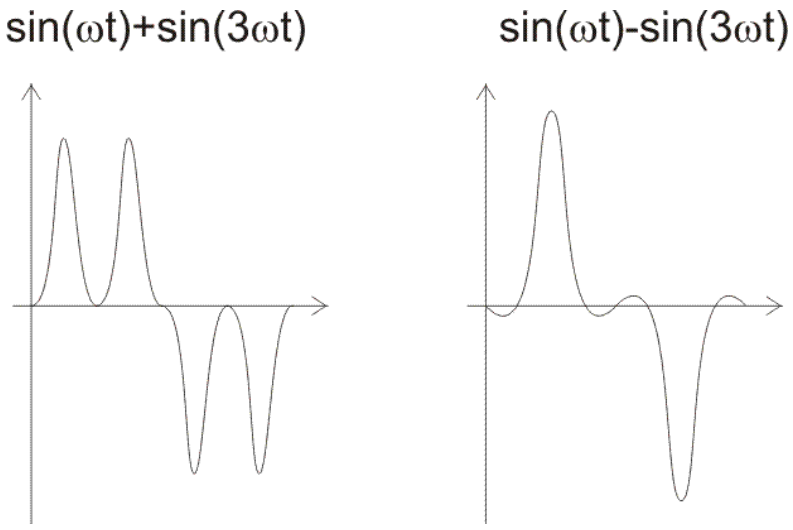


Fig.3.1. Illustration of input and output signals of non-linear phase systems.

The system introduces a phase shift of 0 radians at the frequency of ω, and π radians at three times that frequency. Input signal consists of natural frequency ω and one harmonic with the same amplitude at three times that frequency. Figure 2-1. shows the block diagram of input signal (left) and output signal (right). It is obvious that these two signals have different waveforms. The power of signals is not changed, nor the amplitudes of harmonics, only the phase of the second harmonic is changed.

If we assume that the input is a speech signal whose phase characteristic is not of the essence, such distortion in the phase of the signal would be unimportant. In this case, the system satisfies all necessary requirements. However, if the phase characteristic is of importance, such a great distortion mustn't be allowed.

In order that the phase characteristic of a FIR filter is linear, the impulse response must be symmetric or anti-symmetric, which is expressed in the following way:

h[n] = h[N-n-1] ; symmetric impulse response (about its middle element)

h[n] = -h[N-n-1] ; anti-symmetric impulse response (about its middle element)

One of the drawbacks of FIR filters is a high order of designed filter. The order of FIR filter is remarkably higher compared to an IIR filter with the same frequency response. This is the reason why it is so important to use FIR filters only when the linear phase characteristic is very important.

A number of delay lines contained in a filter, i.e. a number of input samples that should be saved for the purpose of computing the output sample, determines the order of a filter. For example, if the filter is assumed to be of order 10, it means that it is necessary to save 10 input samples preceeding the current sample. All eleven samples will affect the output sample of FIR filter.

The transform function of a typical FIR filter can be expressed as a polynomial of a complex variable z^{-1}. All the poles of the transfer function are located at the origin. For this reason, FIR filters are guaranteed to be stable, whereas IIR filters have potential to become unstable.

Finite impulse response (FIR) filter design methods

Most FIR filter design methods are based on ideal filter approximation. The resulting filter approximates the ideal characteristic as the filter order increases, thus making the filter and its implementation more complex.

The filter design process starts with specifications and requirements of the desirable FIR filter. Which method is to be used in the filter design process depends on the filter specifications and implementation. This chapter discusses the FIR filter design method using window functions.

Each of the given methods has its advantages and disadvantages. Thus, it is very important to carefully choose the right method for FIR filter design. Due to its simplicity and efficiency, the window method is most commonly used method for designing filters. The sampling frequency method is easy to use, but filters designed this way have small attenuation in the stopband.

As we have mentioned above, the design process starts with the specification of desirable FIR filter.

Basic concepts and FIR filter specification

First of all, it is necessay to learn the basic concepts that will be used further in this book. You should be aware that without being familiar with these concepts, it is not possible to understand analyses and synthesis of digital filters.

Figure 3.2 illustrates a low-pass digital filter specification. The word specification actually refers to the frequency response specification.

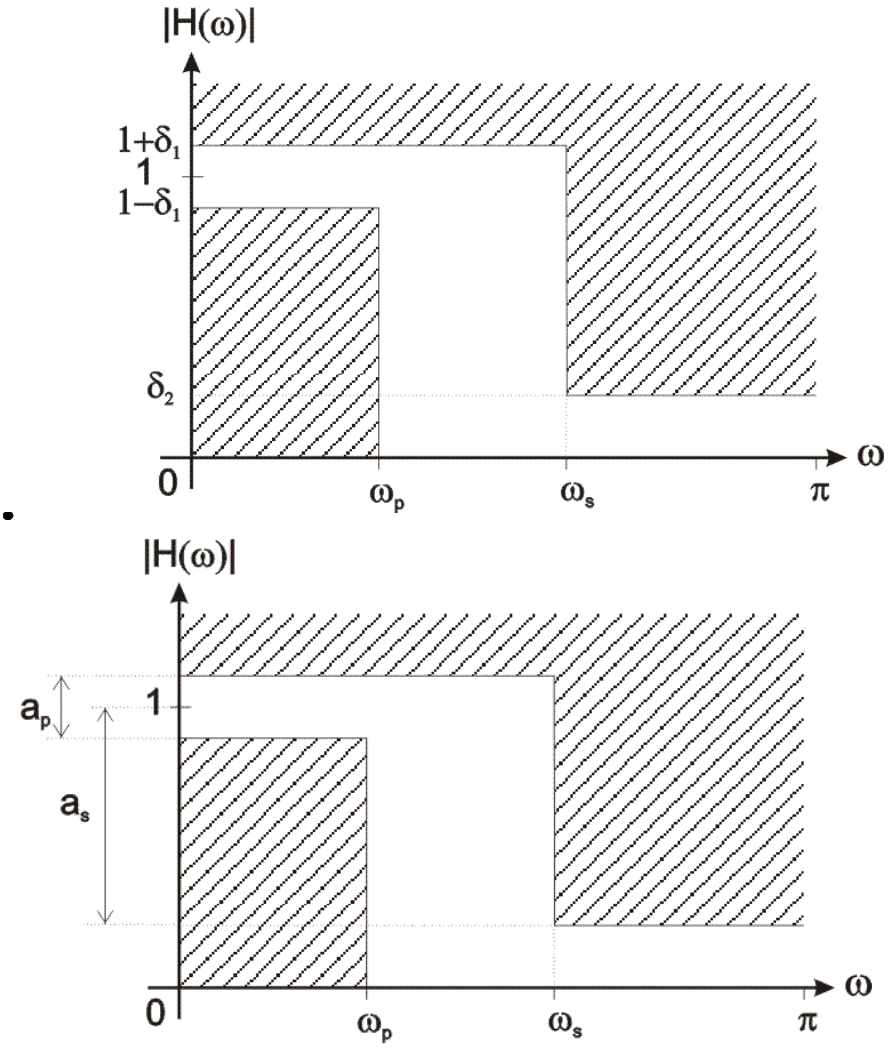


Fig.3.2. A low-pass digital filter specification

- **ωp – normalized cut-off frequency in the passband;**
- **ωs – normalized cut-off frequency in the stopband;**
- **δ1 – maximum ripples in the passband;**
- **δ2 – minimum attenuation in the stopband [dB];**

- **ap – maximum ripples in the passband; and**
- **as – minimum attenuation in the stopband [dB].**

$$a_p = 20 \log_{10}\left(\frac{1+\delta_1}{1-\delta_1}\right)$$

$$a_s = -20 \log_{10} \delta_2$$

Frequency normalization can be expressed as follows:

$$\omega = \frac{2\pi f}{f_s}$$

where:

- **fs is a sampling frequency;**
- **f is a frequency to normalize; and**
- **ω is normalized frequency.**

Table.3.1.Filters

Type of filter	Frequency response $h_d[n]$
low-pass filter	$h_d[n] = \begin{cases} \frac{\sin[\omega_c(n-M)]}{\pi(n-M)}; & n \neq M \\ \frac{\omega_c}{\pi}; & n = M \end{cases}$
high-pass filter	$h_d[n] = \begin{cases} 1-\frac{\omega_c}{\pi}; & n \neq M \\ -\frac{\sin(\omega_c(n-M))}{\pi(n-M)}; & n = M \end{cases}$
band-pass filter	$h_d[n] = \begin{cases} \frac{\sin(\omega_{c2}(n-M))}{\pi(n-M)} - \frac{\sin(\omega_{c1}(n-M))}{\pi(n-M)}; & n \neq M \\ \frac{\omega_{c2}-\omega_{c1}}{\pi}; & n = M \end{cases}$
band-stop filter	$h_d[n] = \begin{cases} \frac{\sin(\omega_{c1}(n-M))}{\pi(n-M)} - \frac{\sin(\omega_{c2}(n-M))}{\pi(n-M)}; & n \neq M \\ 1-\frac{\omega_{c2}-\omega_{c1}}{\pi}; & n = M \end{cases}$

The value of variable n ranges between 0 and N, where N is the filter order. A constant M can be expressed as M = N / 2. Equivalently, N can be expressed as N = 2M.

The constant M is an integer if the filter order N is even, which is not the case with odd order filters. If M is an integer (even filter order), the ideal filter frequency response is symmetric about its Mth sample which is found via expression shown in the table 2-2-1 above. If M is not an integer, the ideal filter frequency response is still symmetric, but not about some frequency response sample.

Since the variable n ranges between 0 and N, the ideal filter frequency response has N+1 sample.

If it is needed to find frequency response of a non-standard ideal filter, the expression for inverse Fourier transform must be used:

$$h_d[n] = \frac{1}{\pi}\int_0^{\pi} e^{j\omega(n-M)}d\omega$$

Non-standard filters are rarely used. However, if there is a need to use some of them, the integral above must be computed via various numerical methodes.

FIR filter design using window functions

The FIR filter design process via window functions can be split into several steps:

1. **Defining filter specifications;**
2. **Specifying a window function according to the filter specifications;**
3. **Computing the filter order required for a given set of specifications;**
4. **Computing the window function coefficients;**
5. **Computing the ideal filter coefficients according to the filter order;**
6. **Computing FIR filter coefficients according to the obtained window function and ideal filter coefficients;**
7. **If the resulting filter has too wide or too narrow transition region, it is necessary to change the filter order by increasing or decreasing it according to needs, and after that steps 4, 5 and 6 are iterated as many times as needed.**

The final objective of defining filter specifications is to find the desired normalized frequencies (ωc, ωc1, ωc2), transition width and stopband attenuation. The window function and filter order are both specified according to these parameters.

Accordingly, the selected window function must satisfy the given specifications. After this step, that is, when the window function is known, we can compute the filter order required for a given set of specifications. When both the window function and filter order are known, it is possible to calculate the window function coefficients w[n] using the formula for the specified window function.

1. **Rectangular Window: The rectangular window is what you would obtain if you were to simply segment a finite portion of the impulse response without any shaping in the time domain:**

$$w(n) = 1 \quad 0 \le n \le M,$$
$$= 0 \text{ otherwise}$$

2. **Hanning window**

The Hanningwindow(or more properly, the von Hann window) is nothing more than a raised cosine:

$$w(n) = 0.5 - 0.5\cos\left(\frac{2\pi n}{M}\right) \quad 0 \le n \le M,$$

$$=0 \text{ otherwise}$$

3. **Hamming window**

$$w(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{M}\right) \quad 0 \le n \le M,$$

$$=0 \text{ otherwise}$$

After estimating the window function coefficients, it is necessary to find the ideal filter frequency samples. The expressions used for computing these samples are discussed in section 2.2.3 under Ideal filter approximation. The final objective of this step is to obtain the coefficients $h_d[n]$. Two sequencies w[n] and $h_d[n]$ have the same number of elements.

The next step is to compute the frequency response of designed filter h[n] using the following expression:

$$h[n] = w[n] \cdot h_d[n]$$

Lastly, the transfer function of designed filter will be found by transforming impulse response via Fourier transform:

$$H(e^{j\omega}) = \sum_{n=0}^{N} h[n] \cdot e^{-jn\omega}$$

or via Z-transform:

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

If the transition region of designed filter is wider than needed, it is necessary to increase the filter

order, reestimate the window function coefficients and ideal filter frequency samples, multiply them in order to obtain the frequency response of designed filter and reestimate the transfer function as well. If the transition region is narrower than needed, the filter order can be decreased for the purpose of optimizing hardware and/or software resources. It is also necessary to reestimate the filter frequency coefficients after that.

PROBLEMS

Use the window design method to design a linear phase FIR filter of order N = 24 to approximate the following ideal frequency response magnitude

$$|H_d(e^{j\omega})| = \begin{cases} 1 & |\omega| \le 0.2\pi \\ 0 & 0.2\pi < |\omega| \le \pi \end{cases}$$

The ideal filter that we would like to approximate is a low-pass filter with a cutoff frequency = 0.2. With N = 24, the frequency response of the filter that is to be designed has the form

$$H(e^{j\omega}) = \sum_{n=0}^{24} h(n)e^{-jn\omega}$$

Therefore, the delay of $h(n)$ is = $N/2$ = 12, and the ideal unit sample response that is to be windowed is

$$h_d(n) = \frac{\sin[0.2\pi(n-12)]}{(n-12)\pi}$$

All that is left to do in the design is to select a window. With the length of the window fixed, there is a trade-off between the width of the transition band and the amplitude of the passband and stopband ripple. With a rectangular window, which provides the smallest transition band,

and the filter is

$$\Delta\omega = 2\pi \cdot \frac{0.9}{24} = 0.075\pi$$

$$h(n) = \begin{cases} \dfrac{\sin[0.2\pi(n-12)]}{(n-12)\pi} & 0 \le n \le 24 \\ 0 & \text{otherwise} \end{cases}$$

However, the stopband attenuation is only 21 dB, which is equivalent to a ripple of 0.089. With a Hamming window, on the other hand,

$$h(n) = \left[0.54 - 0.46\cos\left(\frac{2\pi n}{24}\right)\right] \cdot \frac{\sin[0.2\pi(n-12)]}{(n-12)\pi} \qquad 0 \le n \le 24$$

and the stopband attenuation is 53 dB, or ? $_s$ = 0.0022. However, the width of the transition band increases to

$$\Delta\omega = 2\pi \cdot \frac{3.3}{24} = 0.275\pi$$

which, for most designs, would be too wide.

Frequency sampling method:

The frequency sampling method allows us to design recursive and nonrecursive IIR filters for both standard frequency selective and filters with arbitrary frequency response. A. No recursive frequency sampling filters : The

problem of FIR filter design is to find a finite– length impulse response h (n) that corresponds to desired frequency response. In this method h (n) can be determined by uniformly sampling, the desired frequency response $H_D(\omega)$ at the N points and finding its inverse DFT of the frequency samples.

Design of Optimum Equiripple Linear-Phase FIR

The window method and the frequency-sampling method are relatively simple techniques for designing linear-phase FIR filters. However, they also possess some minor disadvantages, , which may render them undesirable for some applications. A major problem is the lack of precise control of theoritical frequencies such *ws.*The filter design method described in this section is formulated as a Chebyshev approximation problem . It is viewed as an optimum design criterion in the sense that the weighted approximation error between the desired frequency response and the actual frequency response is spread evenly across the pass-band and evenly across the stopband of the filter minimizing the maximum error. The resulting filter designs have ripples in both the passband and the stop-band.To describe the design procedure, let us consider the design of a low-pass filter with pass-band edge frequency *a>p* and stopband edge frequency .

Structure realization of FIR Filters
In signal processing, a digital filter is a system that performs mathematical operations on a sampled, discrete-time signal to reduce or enhance certain aspects of that signal. This is in contrast to the other major type of electronic filter, the analog filter, which is an electronic circuit operating on continuous-time analog signals.

A digital filter system usually consists of an analog-to-digital converter to sample the input signal, followed by a microprocessor and some peripheral components such as memory to store data and filter coefficients etc. Finally a digital-to- analog converter to complete the output stage. Program Instructions (software) running on the microprocessor implement the digital filter by performing the necessary mathematical operations on the numbers received from the ADC. In some high performance applications, an FPGA orASIC is used instead of a general purpose microprocessor, or a specialized DSP with specific paralleled architecture for expediting operations such as filtering.

Digital filters may be more expensive than an equivalent analog filter due to their increased complexity, but they make practical many designs that are impractical or impossible as analog filters. When used in the context of real-time analog systems, digital filters sometimes have problematic latency (the difference in time between the input and the response) due to the associated analog-to-digital and digital-to- analog conversions and anti-aliasing filters, or due to other delays in their implementation.

Digital filters are commonplace and an essential element of everyday electronics such as radios, cellphones, and AV receivers.

Characterization

A digital filter is characterized by its transfer function, or equivalently, its difference equation. Mathematical analysis of the transfer function can describe how it will respond to any input. As such, designing a filter consists of developing specifications appropriate to the problem (for example, a second-order low pass filter with a specific cut-off frequency), and then producing a transfer function which meets the specifications.

The transfer function for a linear, time-invariant, digital filter can be expressed as a transfer function in the *Z*-domain; if it is causal, then it has the form:

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_N z^{-N}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_M z^{-M}}$$

where the order of the filter is the greater of *N* or *M*. See *Z*-transform's LCCD equation for further discussion of this transfer function.

This is the form for a recursive filter with both the inputs (Numerator) and outputs (Denominator), which typically

leads to an IIR infinite impulse response behaviour, but if the denominator is made equal to unity i.e. no feedback, then this becomes an FIR or finite impulse response filter.

The impulse response, often denoted $h(k)$ or h_k, is a measurement of how a ilter will respond to the Kronecker delta function. Digital filters are typically considered in two categories: infinite impulse response (IIR) and finite impulse response (FIR). In the case of linear time-invariant FIR filters, the impulse response is exactly equal to the sequence of filter coefficients:

IIR filters on the other hand are recursive, with the output depending on both current and previous inputs as well as

$$y_n = \sum_{k=0}^{n-1} h_k x_{n-k}$$

previous
outputs. The general form of an IIR filter is thus:

$$\sum_{m=0}^{M-1} a_m y_{n-m} = \sum_{k=0}^{n-1} b_k x_{n-k}$$

Plotting the impulse response will reveal how a filter will respond to a sudden, momentary disturbance.

1. Difference equation

In discrete-time systems, the digital filter is often implemented by converting the transfer function to a linear constant- coefficient difference equation (LCCD) via the Z-transform. The discrete frequency-domain transfer function is written as the ratio of two polynomials. For example:

This is expanded:

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

$$H(z) = \frac{z^2+2z+1}{z^2+\frac{1}{4}z-\frac{3}{8}}$$

and to make the corresponding filter causal, the numerator and denominator are divided by the highest order of z:

$$H(z) = \frac{1+2z^{-1}+z^{-2}}{1+\frac{1}{4}z^{-1}-\frac{3}{8}z^{-2}} = \frac{Y(z)}{X(z)}$$

The coefficients of the denominator, a_k, are the 'feed-backward' coefficients and the coefficients of the numerator are the 'feed-forward' coefficients, b_k. The resultant linear difference equation is:

$$y[n] = -\sum_{k=1}^{M} a_k y[n-k] + \sum_{k=0}^{N} b_k x[n-k]$$

or, for the example above:

$$\frac{Y(z)}{X(z)} = \frac{1+2z^{-1}+z^{-2}}{1+\frac{1}{4}z^{-1}-\frac{3}{8}z^{-2}}$$

rearranging terms:

$$\Rightarrow (1+\frac{1}{4}z^{-1}-\frac{3}{8}z^{-2})Y(z) = (1+2z^{-1}+z^{-2})X(z)$$

then by taking the inverse *z*-transform:

$$\Rightarrow y[n] + \frac{1}{4}y[n-1] - \frac{3}{8}y[n-2] = x[n] + 2x[n-1] + x[n-2]$$

and finally, by solving for $y[n]$:

$$y[n] = -\frac{1}{4}y[n-1] + \frac{3}{8}y[n-2] + x[n] + 2x[n-1] + x[n-2]$$

This equation shows how to compute the next output sample, , in terms of the past outputs, $y[n-p]$, the present input, $x[n]$, and the past inputs. Applying the filter to an input in this form is equivalent to a Direct Form I or II realization, depending on the exact order of evaluationAfter a filter is designed, it must be *realized* by developing a signal flow diagram that describes the filter in terms of operations on sample sequences.

A given transfer function may be realized in many ways. Consider how a simple expression such as $ax + bx + c$ could be evaluated – one could also compute the equivalent $x(a+b) + c$. In the same way, all realizations may be seen as "factorizations" of the same transfer function, but different realizations will have different numerical properties. Specifically, some realizations are more efficient in terms of the number of operations or storage elements required for their implementation, and others provide advantages such as improved numerical stability and reduced round-off error. Some structures are better for fixed-point arithmetic and others may be better for floating-point arithmetic.

1. Direct Form I

A straightforward approach for IIR filter realization is Direct Form I, where the difference equation is evaluated directly. This form is practical for small filters, but may be inefficient and impractical (numerically unstable) for complex designs.[3] In general, this form requires 2N delay elements (for both input and output signals) for a filter of order N.

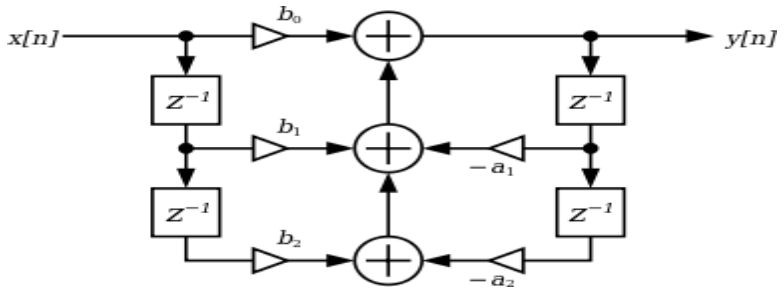


Fig 3.3. Direct form I

Direct Form II

The alternate Direct Form II only needs *N* delay units, where *N* is the order of the filter – potentially half as much as Direct Form I. This structure is obtained by reversing the order of the numerator and denominator sections of Direct Form I, since they are in fact two linear systems, and the commutativity property applies. Then, one will notice that there are two columns of delays (z^{-1}) that tap off the center net, and these can be combined since they are redundant, yielding the implementation as shown below.

The disadvantage is that Direct Form II increases the possibility of arithmetic overflow for filters of high Q or resonance.[4] It has been shown that as Q increases, the round-off noise of both direct form topologies increases without bounds.[5] This is because, conceptually, the signal is first passed through an all-pole filter (which normally boosts gain at the resonant frequencies) before the result of that is saturated, then passed through an all-zero filter (which often attenuates much of what the all-pole half amplifies).

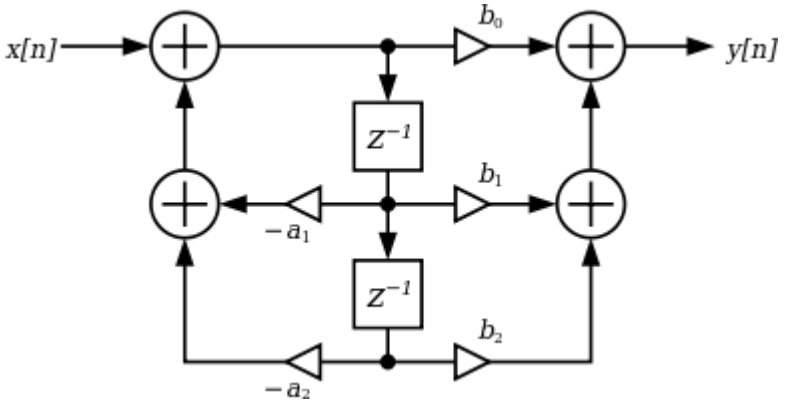


Fig.3.4. Direct form II

2. Cascaded second-order sections

A common strategy is to realize a higher-order (greater than 2) digital filter as a cascaded series of second-order "biquadratric" (or "biquad") (see digital biquad filter). The advantage of this strategy is that the coefficient range is limited.

Cascading direct form II sections results in N delay elements for filters of order N. Cascading direct form I sections results in N+2 delay elements since the delay elements of the input of any section (except the first section) are redundant with the delay elements of the output of the preceding section.

3. Linear-Phase FIR Structures Phase FIR Structures

The symmetry (or antisymmetry) property of a linear-phase FIR filter can be exploited to reduce the number of multipliers into almost half of that in the direct form implementations

- **Consider a length-7 Type 1 FIR transfer function with a symmetric impulse response:** $H(Z) = h(0) + h(1)Z^{-1} + h(2)Z^{-2} + h(3)Z^{-3} + h(2)Z^{-4} + h(1)Z^{-5} + h(0)Z^{-6}$ **.Rearranging, we get**

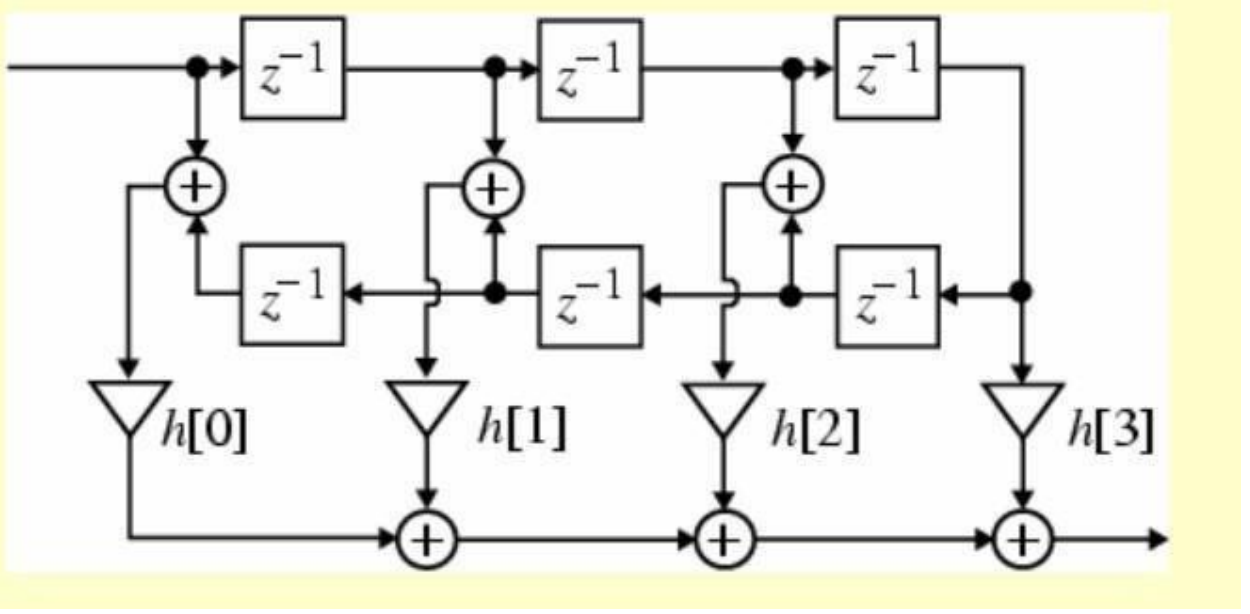


Fig.3.5. Linear phase FIRI

4. PolyphasePolyphase FIR Structures FIR Structures

The polyphase decomposition of H(z) leads to a parallel form structure.

To illustrate this approach, consider a causal FIR transfer function H(z) with N = 8:

$$H(Z) = h(0) + h(1)Z^{-1} + h(2)Z^{-2} + h(3)Z^{-3} + h(4)Z^{-4} + h(5)Z^{-5} + h(6)Z^{-6} + h(7)Z^{-7} + h(8)Z^{-8}$$

H(z) can be expressed as a sum of two terms, with one term containing the even indexed coefficients and the other containing the odd-indexed

$$H(Z) = h(0) + h(2)Z^{-2} + h(4)Z^{-4} + h(6)Z^{-6} + h(8)Z^{-8} + Z^{-1}[h(1) + h(3)Z^{-2} + h(5)Z^{-4} + h(7)Z^{-6}]$$

Putting $H(Z)$

$$= E_0(Z^2) + Z^{-1}E_1(Z^2)$$

The subfilters in the polyphase realization of an FIR transfer function are also FIR filters and can be realized using any methods. However, to obtain a canonic realization of the overall structure, the delays in all subfilters must be shared.

The filters designed by considering all the infinite samples of impulse response are called IIR (Infinite Impulse Response) filters. In digital domain, the processing of infinite samples of impulse response is practically not possible. Hence direct design of IIR filter is not possible. Therefore, the IIR filters are designed via analog filters. In design of IIR filter, the specification of an IIR filter is transformed to specification of an analog filter and an analog filter with transfer function, H(s) is designed to satisfy the specification. Then the analog filter is transformed to digital filter with transfer function, H(z). We know that the analog filter with transfer function H(s) is stable if all its poles lie in the left half of the s-plane. Consequently, if the conversion technique is to be effective, it should possess the following desirable properties. 1. The imaginary axis in the s-plane should map into the unit circle in

the z-plane. Thus there will be a direct relationship between the two frequency variables in the two domains. 2. The left-half of the s- plane should map into the interior of the unit circle in the z-plane. Thus a stable analog filter will

be converted to a stable digital filter. The analog filter is designed by approximating the ideal frequency response using an error function. A number of solutions to the approximation problem of analog filter design are well developed. The popular among them are Butterworth and Chebyshev approximation. The popular transformation techniques used for transforming analog filter transfer function H(s) to digital filter transfer function H(z) are bilinear and impulse invariant transformation. The digital transfer function H(z) can be realized in a software that runs on a digital hardware (or it can be implemented in firmware). The frequency response $H(e^{jw})$ by letting z = ejw in the transfer function H(z) of the filter.

5. Design of IIR filters

The ideal magnitude response, $|H_d(jW)|$ of the four basic types of analog filters are shown in fig (a), (b), (c) and (d). The ideal magnitude response has sudden transition from passband to stopband which is practically not realizable. Hence the ideal response is approximated using a filter approximation function. The approximation problem is solved to meet a specified tolerance in the passband and stopband. The shaded areas in the fig 7.1 shows the tolerance regions of the ideal frequency response. In the passband the magnitude is approximated to unity within an error of d_p . In the stopband the magnitude is approximated to zero within an error of ds . Here the dp and ds are the limits of the tolerance in the passband and stopband. The dp and ds are also called ripples. The frequency repsonse of practical analog filter shows edges for passband and stopband so that the tolerances are within specified limits. Now, the specification of practical analog filter will be the following. W_p = Passband edge frequency in rad/second. W_s = Stopband edge frequency in rad/second. A_p = Gain at passband edge frequency A_s = Gain at stopband edge frequency.

Frequency selective filters: Ideal filter characteristics

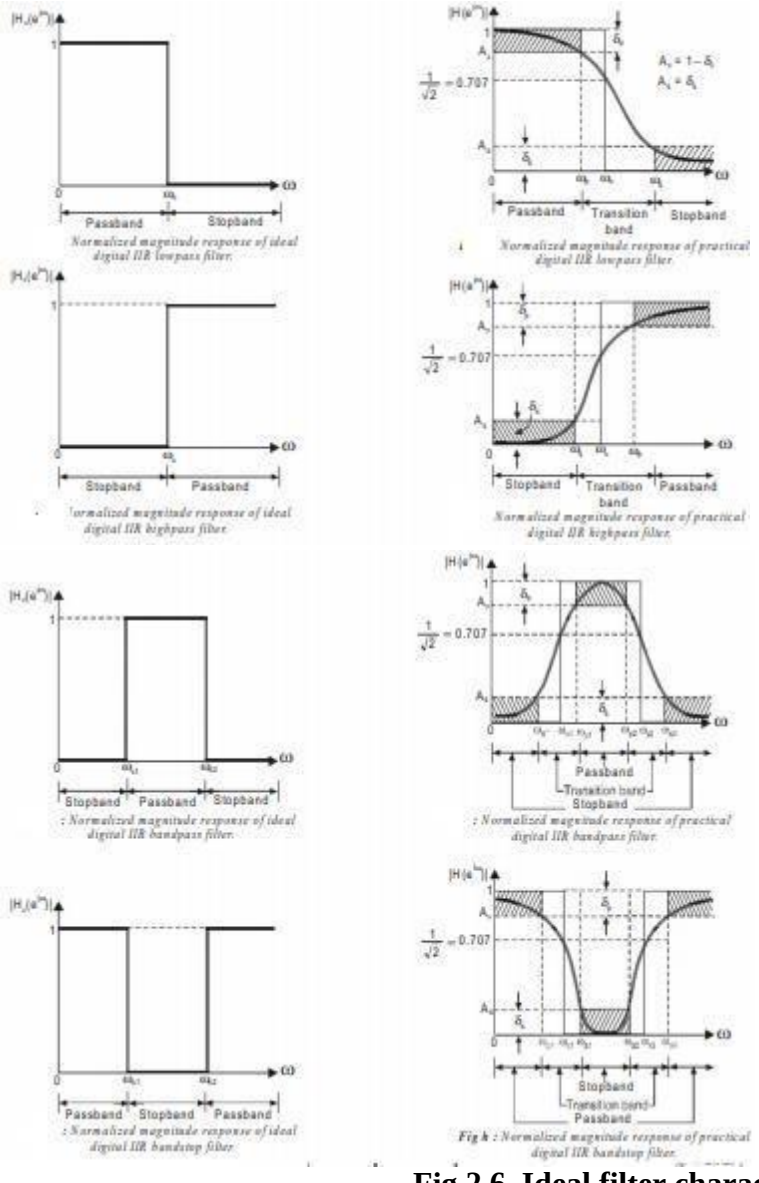


Fig.2.6. Ideal filter characteristics

Impulse Invariant Transformation

The objective of impulse invariant transformation is to develop an IIR filter transfer function whose impulse response is the sampled version of the impulse response of the analog filter. The main idea behind this technique is to preserve the frequency response characteristics of the analog filter. It can be stated that the frequency response of digital filter will be identical with the frequency response of the corresponding analog filter if the sampling time period T is selected sufficiently small (or the sampling frequency should be high) to minimize (or avoid completely) the effects of aliasing.

$$\frac{1}{s + p_i} \xrightarrow{\text{(is transformed to)}} \frac{1}{1 - e^{-p_i T} z^{-1}}$$

Relation Between Analog and Digital Frequency in Impulse Invariant Transformation Let, W = Analog frequency in rad/second.

w = Digital frequency in rad/sample

$$\text{Digital frequency, } w = WT \quad \text{or} \quad \text{Analog frequency, } \Omega = \frac{\omega}{T}$$

Thus the mapping from the analog frequency W to the digital frequency w is many-to-one. This reflects the effects of aliasing due to sampling.

Useful Impulse Invariant Transformation

$$\frac{1}{(s + p_i)^m} \longrightarrow \frac{(-1)^{m-1}}{(m-1)!} \frac{d^{m-1}}{dp_i^{m-1}} \frac{1}{1 - e^{-p_i T} z^{-1}}$$

$$\frac{(s + a)}{(s + a)^2 + b^2} \longrightarrow \frac{1 - e^{-aT} (\cos bT) z^{-1}}{1 - 2e^{-aT} (\cos bT) z^{-1} + e^{-2aT} z^{-2}}$$

$$\frac{b}{(s + a)^2 + b^2} \longrightarrow \frac{e^{-aT} (\sin bT) z^{-1}}{1 - 2e^{-aT} (\cos bT) z^{-1} + e^{-2aT} z^{-2}}$$

Bilinear Transformation

The bilinear transformation is a conformal mapping that transforms the imaginary axis of s-plane into the unit circle in the z-plane only once, thus avoiding aliasing of frequency components. In this mapping all points in the left half of s-plane are mapped inside the unit circle in the z-plane and all points in the right half of s-plane are mapped outside the unit circle in the z- plane. The bilinear transformation can be linked to the trapezoidal formula for numerical integration. Any analog system is governed by a differential equation in time domain.

In the s-domain transfer function, if "s" is substituted by the term $\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}$ the resulting transfer function will be z-domain transfer function.

Relation Between Analog and Digital Filter Poles in Bilinear Transformation

The mapping of s-domain function to z-domain function by bilinear transformation is a one to one mapping, that is, for every point in z-plane, there is exactly one corresponding point in s- plane and vice versa. The transformation is accomplished when,

$$s = \frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}$$

$$\therefore \text{ Analog frequency, } \Omega = \frac{2}{T}\tan\frac{\omega}{2} \qquad \therefore \text{ Digital frequency, } \omega = 2\tan^{-1}\frac{\Omega T}{2}$$

Specifications of Digital IIR Lowpass Filter

Let, $H(e^{jw})$ = Frequency response of IIR filter.

$|H(e^{jw})|$ = Magnitude response of IIR filter.

The magnitude response, $|H(e^{jw})|$ of IIR filter will have a passband, transition band and stop band.

The specification of the IIR filter can be expressed in any one of the following three different ways.

Case i : Gain at passband and stopband edge frequency

Case ii : Attenuation at passband and stopband edge frequency

Case iii: Ripple at passband and stopband edge frequency

The gain can be expressed either in normal values or in decibels (dB). The maximum value of normalized gain is unity and so the gain at band edge frequencies will be less than 1. Therefore, the dB-gain will be negative.

Let, w_p = Passband edge digital frequency in rad/sample. w_s =

Stopband edge digital frequency in rad/sample.

$A_p = |H(e^{jw})|_{w = wp}$ = Gain (or magnitude) at passband edge frequency. A_s =

$|H(ejw)|_{w = ws}$ = Gain (or magnitude) at stopband edge frequency.

A_p,dB = 20 log $[|H(ejw)|_{w = wp}]$ = dB-Gain (or dB-magnitude) at passband edge frequency.

A_s,dB = 20 log $[|H(ejw)|_{w = ws}]$ = dB-Gain (or dB-magnitude) at stopband edge frequency.

The gain in normal values can be converted to dB-gain or vice versa as shown below.

$$A_p,dB = 20 \log Ap$$

$$A_p = 10^{(Ap,dB/20)}$$

$$A_s,dB = 20 \log A_s$$

$$A_s = 10^{(As,dB/20)}$$

The attenuation is usually expressed in decibels (dB). Since the gain at edge frequencies are less than 1, the attenuation in normal values will be greater than 1, and the dB-attenuation is positive.

$$\text{Let, } \alpha_p = \frac{1}{A_p} = \frac{1}{\left|H(e^{j\omega})\right|_{\omega=\omega_p}} = \text{Attenuation at passband edge frequency}$$

$$\alpha_s = \frac{1}{A_s} = \frac{1}{\left|H(e^{j\omega})\right|_{\omega=\omega_s}} = \text{Attenuation at stopband edge frequency}$$

$$\alpha_{p,dB} = 20\log\left[\frac{1}{A_p}\right] = 20\log\left[\frac{1}{\left|H(e^{j\omega})\right|_{\omega=\omega_p}}\right] = \text{dB-Attenuation at passband edge frequency}$$

$$\alpha_{s,dB} = 20\log\left[\frac{1}{A_s}\right] = 20\log\left[\frac{1}{\left|H(e^{j\omega})\right|_{\omega=\omega_s}}\right] = \text{dB-Attenuation at stopband edge frequency}$$

Ripple at passband and stopband edge frequency:

$$A_p = 1 - \delta_p \qquad \alpha_p = \frac{1}{A_p} = \frac{1}{1-\delta_p}$$

$$A_s = \delta_s \qquad \alpha_s = \frac{1}{A_s} = \frac{1}{\delta_s}$$

6. Transfer function of Analog Butterworth Lowpass Filter:

The analog filter transfer function of normalized and unnormalized butterworth lowpassfiltersare givenbelow.Let, Nbetheorder ofthefilter. Let, H(sn) bethe normalized Butterworth lowpassfilter transfer function. When N is even

$$H(s_n) = \prod_{k=1}^{\frac{N}{2}} \frac{1}{s_n^2 + b_k s_n + 1}$$

When N is odd,

$$H(s_n) = \frac{1}{s_n + 1} \prod_{k=1}^{\frac{N-1}{2}} \frac{1}{s_n^2 + b_k s_n + 1}$$

$$\text{where}, b_k = 2 \sin\left[\frac{(2k-1)\pi}{2N}\right]$$

Table. Summary of Butterworth Lowpass Filter Normalized Transfer Function

Order, N	Normalized tansfer function, $H(s_n)$
1	$\frac{1}{s_n+1}$
2	$\frac{1}{s_n^2+1.414s_n+1}$
3	$\frac{1}{(s_n+1)(s_n^2+s_n+1)}$
4	$\frac{1}{(s_n^2+0.765s_n+1)(s_n^2+1.848s_n+1)}$
5	$\frac{1}{(s_n+1)(s_n^2+0.618s_n+1)(s_n^2+1.618s_n+1)}$
6	$\frac{1}{(s_n^2+1.932s_n+1)(s_n^2+1.414s_n+1)(s_n^2+0.518s_n+1)}$

7. Order of the Lowpass Butterworth Filter

In Butterworth filters the frequency response of the filter depends on the order, N. Hence the order N has to be estimated to satisfy the given specifications. Usually the specifications of the filter are given in terms of gain at a passband and stopband frequency. Let, A_p = Gain or Magnitude at a passband frequency W_p.

A_s = Gain or Magnitude at a stopband frequency W_s.

$$N_1 = \frac{1}{2}\frac{\log\left[\frac{(1/A_s^2)-1}{(1/A_p^2)-1}\right]}{\log\left(\frac{\Omega_s}{\Omega_p}\right)} \qquad N_1 = \frac{\log\left[\left(\frac{10^{0.1\alpha_{s,dB}}-1}{10^{0.1\alpha_{p,dB}}-1}\right)^{\frac{1}{2}}\right]}{\log\frac{\Omega_s}{\Omega_p}}$$

$$\text{Cutoff frequency, } \Omega_c = \frac{\Omega_s}{\left[(1/A_s^2)-1\right]^{\frac{1}{2N}}}$$

Alternatively,

$$\text{Cutoff frequency, } \Omega_c = \frac{\Omega_p}{\left[(1/A_p^2)-1\right]^{\frac{1}{2N}}}$$

For bilinear transformation,

$$\Omega_p = \frac{2}{T}\tan\frac{\omega_p}{2} \quad ; \quad \Omega_s = \frac{2}{T}\tan\frac{\omega_s}{2}$$

For impulse invariant transformation,

$$\Omega_p = \frac{\omega_p}{T} \quad ; \quad \Omega_s = \frac{\omega_s}{T}$$

where T is the sampling time.

$$\text{Cutoff frequency, } \Omega_c = \frac{\Omega_s}{\left(10^{0.1\alpha_{s,dB}}-1\right)^{\frac{1}{2N}}}$$

Alternatively,

$$\text{Cutoff frequency, } \Omega_c = \frac{\Omega_p}{\left(10^{0.1\alpha_{p,dB}}-1\right)^{\frac{1}{2N}}}$$

8. Design Procedure for Lowpass Digital Butterworth IIR Filter

- **The process of filter design begins with filter specifications which include the filter characteristics (Lowpass, high-pass, band-pass, band-stop filter), filter type, passband frequency, stopband frequency, transistion width frequency, sampling frequency and filter length.)**
- **The second step is obtain filter response, H(ω)**
- **Third step is to find the filter coefficient and acceptable filter.**
- **The last step is to implement filter coefficient and choose ω appropriate filter**

structure for filter implementation.

- **There are 2 commons IIR filter design**
- **1. Butterworth (As the Filter Order, N increases, the transition band becomes narrower).**
- **2. Chebyshev Type**
- **The analog filter will be mapped to digital filter using transformation of s-domain to z- domain. 2 methods to convert the analog filter to digital filter and vice versa;**
- **1. Impulse Invariance method**

2. Bilinear Transformation method

1. Choose either bilinear or impulse invariant transformation, and determine the specifications of equivalent analog filter. The gain or attenuation of analog filter is same as digital filter. The band edge frequencies are calculated using the following equations.

Let, W_p = Passband edge analog frequency corresponding to w_p.

W_s = Stopband edge analog frequency corresponding to w_s.

For bilinear transformation,

$$\Omega_p = \frac{2}{T}\tan\frac{\omega_p}{2}$$

$$\Omega_s = \frac{2}{T}\tan\frac{\omega_s}{2}$$

Note : If either T or F_s is not specified then take T= 1 second.

If F_s is specified, then $T = \frac{1}{F_s}$

For impulse invariant transformation,

$$\Omega_p = \frac{\omega_p}{T}$$

$$\Omega_s = \frac{\omega_s}{T}$$

2. Decide the order N of the filter. In order to estimate the order N, calculate a parameter N_1 using the following equation.

$$N_1 = \frac{1}{2} \frac{\log\left[\frac{\left(1/A_s^2\right)-1}{\left(1/A_p^2\right)-1}\right]}{\log\left(\frac{\Omega_s}{\Omega_p}\right)}$$

Choose N such that, $N \geq N_1$. Usually N is chosen as nearest integer just greater than N_1.

3. Determine the normalized transfer function, $H(s_n)$ of the analog lowpass filter.

When N is even,

$$H(s_n) = \prod_{k=1}^{\frac{N}{2}} \frac{1}{s_n^2 + b_k s_n + 1}$$

When N is odd,

$$H(s_n) = \frac{1}{s_n + 1} \prod_{k=1}^{\frac{N-1}{2}} \frac{1}{s_n^2 + b_k s_n + 1}$$

where, $b_k = 2\sin\left[\frac{(2k-1)\pi}{2N}\right]$

4. Calculate the analog cutoff frequency, Ω_c.

$$\text{Cutoff frequency, } \Omega_c = \frac{\Omega_s}{\left[\left(1/A_s^2\right)-1\right]^{\frac{1}{2N}}}$$

5. Determine the unnormalized analog transfer function H(s) of the lowpass filter.

$$H(s) = H(s_n)\Big|_{s_n = \frac{s}{\Omega_c}}$$

When the order N is even, H(s) is obtained by letting $s_n \rightarrow s/\Omega_c$ in equation (7.58).

$$\therefore H(s) = \prod_{k=1}^{\frac{N}{2}} \frac{1}{s_n^2 + b_k s_n + 1}\Bigg|_{s_n = \frac{s}{\Omega_c}} = \prod_{k=1}^{\frac{N}{2}} \frac{\Omega_c^2}{s^2 + b_k \Omega_c s + \Omega_c^2}$$

When the order N is odd, H(s) is obtained by letting $s_n \rightarrow s/\Omega_c$ in equation (7.59).

$$\therefore H(s) = \frac{1}{s_n + 1} \prod_{k=1}^{\frac{N-1}{2}} \frac{1}{s_n^2 + b_k s_n + 1}\Bigg|_{s_n = \frac{s}{\Omega_c}} = \frac{\Omega_c}{s + \Omega_c} \prod_{k=1}^{\frac{N-1}{2}} \frac{\Omega_c^2}{s^2 + b_k \Omega_c s + \Omega_c^2}$$

6. Determine thetransfer functionofdigitalfilter, H(z). Usingthechosen transformationinstep-1,transformH(s)toH(z).Whenimpulseinvariant transformationisemployed, ifT<1,thenmultiply H(z)byTtonormalizethe magnitude.

7. Realize the digital filter transfer function H(z) by a suitable structure. 8. Verify the design by sketching the frequency response H(e^{jw}).

$$H(e^{j\omega}) = H(z)\Big|_{z=e^{j\omega}}$$

10. Design of Lowpass Digital Chebyshev Filter

The analog Chebyshev filter is designed by approximating the ideal frequency response using an error function. The approximation function is selected such that the error is minimized over a prescribed band of frequencies.

1. Choose either bilinear or impulse invariant transformation, and determine the specifications of equivalent analog filter. The gain or attenuation of analog filter is same as digital filter. The band edge frequencies are calculated using the following equations.

Let, W_p = Passband edge analog frequency corresponding to w_p.

W_s = Stopband edge analog frequency corresponding to w_s.

For bilinear transformation,

$$\Omega_p = \frac{2}{T}\tan\frac{\omega_p}{2}$$

$$\Omega_s = \frac{2}{T}\tan\frac{\omega_s}{2}$$

Note : If either T or F_s is not specified then take T= 1 sec.

If F_s is specified, then $T = \frac{1}{F_s}$

For impulse invariant transformation,

$$\Omega_p = \frac{\omega_p}{T}$$

$$\Omega_s = \frac{\omega_s}{T}$$

2. Decide the order N of the filter. In order to estimate the order N, calculate a parameter N_1 using the following equation. Choose N such that $N \geq N_1$. Usually N is chosen as nearest integer just greater than N_1.

$$N_1 = \frac{\cosh^{-1}\left[\left(\frac{(1/A_s^2)-1}{(1/A_p^2)-1}\right)^{\frac{1}{2}}\right]}{\cosh^{-1}\left(\frac{\Omega_s}{\Omega_p}\right)}$$

1 Determine the normal ized refer liinction I I(say, of the filter. When the order N is even.

$$H(s_n) \quad \prod_{k=1}^{\hat{} } \frac{B_k}{s_n^{\hat{}} + b_k s_n + c_k}$$

Wben tbe order N i.‹ o‹lA,

$$H(s) \quad \frac{B_0}{s + c_0} \prod_{k=1}^{2} \frac{B_k}{s^2 + b_k s + c_k}$$

where, $b_k = 2\, y_N \sin\left(\frac{(2k-1)\pi}{2N}\right)$

$$c_k = y_N^2 + \cos^2\left(\frac{(2k-1)\pi}{2N}\right)$$

$$y_N \quad \frac{1}{2}\left\{\left[\left(\frac{1}{\epsilon^2}+1\right)^{\overline{2}} + \frac{1}{\epsilon}\right]^{\cdot\cdot} - \left[\left(\frac{1}{\epsilon^2}+1\right)^{\overline{2}} + \frac{1}{\epsilon}\right]\right\}$$

$$\left]^{\overline{2}}\right.$$

For even vaI ties of N. mud B, such iliai.

$$H(0) = \frac{1}{j\,1 + w\,j\,'}$$

For add values o1* N, find B, such that,

Ig0) — I

(It is nomal praciice to iake B, — B, — B, ... BU.

4. Determine theunnormaJLze‹t aoalog ‹fer fuociion Ef(s) of the loa'pass fitier.

$$H(s) = H(s_n)\Big|_{s_n = \frac{s}{\Omega_c}}$$

I lerew , - fl – Passband edge frequency.

When The order N is eveN, I I(s) is obtained by letting s, & sf, in equation7.55).

$$H(sj \quad \tilde{n}_{k\,\cdot\,i} \frac{B_k}{s_n^2 + b_k s_n + c_k} \quad - \prod_{k=1}^{2} \frac{B_k\, \Omega_c^2}{s^2 + b_k\, \Omega_c\, s + c_k\, \Omega_c^2}$$

When the order N ie M, Has} is obtained by letting s 4 in equatiON (7.89}.

$$H(*)= \frac{B_0}{s + c_0} \prod_{k=1}^{2} \frac{B_k}{s_n^2 + b_k s_n + c_k} = \frac{B_n \Omega_c}{s + c_0 \Omega_c} \frac{B_k \Omega_c^2}{s^2 + b_k \Omega_c s + c_k \Omega_c^2}$$

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF ELECTRICAL AND ELECTRONICS

DEPARTMENT OF ELECTRONICS AND COMMMUNICATION ENGINEERING

UNIT - IV
FINITE WORD LENGTH EFFECTS

UNIT 4 FINITE WORD LENGTH EFFECTS

Fixed point and floating point number representations - Comparison - Truncation and Rounding errors - Quantization noise - Derivation for quantization noise power -coefficient quantization error - Product quantization error - Overflow error - Roundoff noise power - limit cycle oscillations due to product round off and overflow errors - signal scaling.

In digital representation the signals are represented as an array of binary numbers, and the digital system employ a fixed size of binary called "word size or word length" for number representation. This finite word size for number representation leads to errors in input signals, intermediate signals in computations and in the final output signals. In general, the various effects due to finite precision representation of numbers in digital systems are called finite word length effects.

Some of the finite word length effects in digital systems are given below.

- **Errors due to quantization of input data.**
- **Errors due to quantization of filter coefficients.**
- **Errors due to rounding the product in multiplication.**
- **Errors due to overflow in addition.**
- **Limit cycles in recursive computations.**

The two major methods of representing binary numbers are fixed point representation and floating point representation.

Fixed point representation the digits allotted for integer part and fraction part are fixed, and so the position of binary point is fixed. Since the number of digits is fixed it is impossible to represent too large and too small numbers by fixed point representation. Therefore the range of numbers that can be represented in fixed point representation for a given binary word size is less when compared to floating point representation.

In fixed point representation there are three different formats for representing negative binary fraction numbers. They are,

1. **Sign-magnitude format**
2. **One's complement format**
3. **Two's complement format**

In sign magnitude format the negative value of a given number differ only in sign bit (i.e., digit d0). The sign digit d0 is zero for positive number and one for negative number.

In one's complement format the negative of the given number is obtained by bit by bit complement of its positive representation.

In two's complement format the negative of the given number is obtained by

taking one's complement of its positive representation and then adding one to the least significant bit.

<u>Floating point representation</u> the binary point can be shifted to desired position so that number of digits in the integer part and fraction part of a number can be varied. This leads to larger range of number that can be represented in floating point representation.

Floating point number, $N_f = M X2^E$

In various digital systems or computers, a variety of formats are employed for floating point representation. The IEEE (Institute of Electrical and Electronic Engineers) has proposed a standard format for floating point representation, which is widely followed in digital computers. The IEEE-754 standard format for 32-bit single precision floating point number is shown in fig

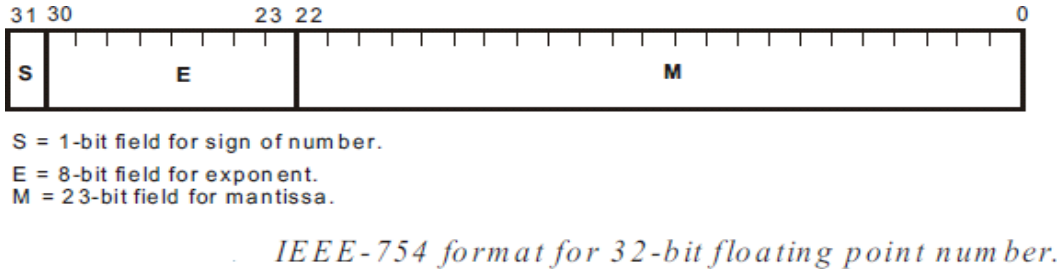


Fig 4.1 IEEE-754 format for 32 bit-floating point

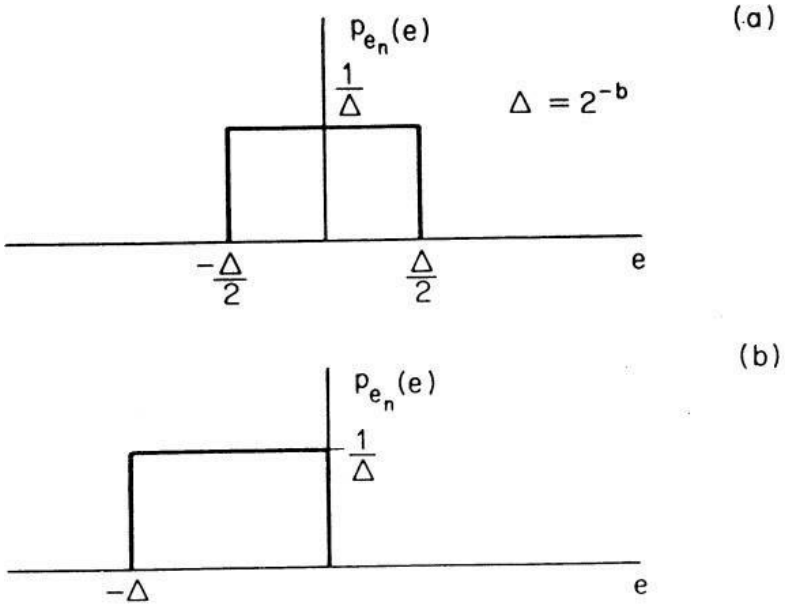
number <u>Comparison of Fixed Point and Floating Point Representation</u>

Fixed point representation	Floating point representation
1. In a b-bit binary the range of numbers represented is less when compared floating point representation.	1. In a b-bit binary the range of numbers represented is large when compared to fixed point representation.
2. The position of binary point is fixed	2. The position of binary point is variable.
3. The resolution is uniform throughout	3. The resolution is variable

<u>Truncation and Rounding error</u>

In fixed point or floating point arithmetic the size of the result of an operation (sum or product) may be exceeding the size of binary used in the number system. In such cases the low order bits has to be eliminated in order to store the result. The two methods of eliminating these low order bits are truncation and rounding. This process is also referred to as quantization via truncation and rounding. The effect of rounding and truncation is to

introduce an error whose value depends on the number of bits eliminated. The characteristics of the errors introduced through either truncation or rounding depend on the type of number representation. The truncation is the process of reducing the size of binary number (or reducing the number of bits in a binary number) by discarding all bits less significant than the least significant bit that is retained. In the truncation of a binary number to b bits, all the less significant bits beyond b^{th} bit are discarded. Rounding is the process of reducing the size of a binary number to finite word size of b-bits such that the rounded b-bit number is closest to the original unquantized number. The rounding process consists of truncation and addition. In rounding of a number to b-bits, first the unquantized number is truncated to b-bits by retaining the most significant b-bits. Then a zero or one is added to the least significant bit of the truncated number depending on the



Probability density functions for (a) rounding; (b) truncation.

bit that is next to the least significant bit that is retained.

Fig 4.2 Probability density function for (a) rounding (b) Truncation

<u>Quantization Steps</u>

The decimal numbers that are encountered as filter coefficients, sum, product, etc., in DSP applications will usually lie in the range of –1 to +1. When "B" bit binary is selected to represent the decimal numbers, then 2^B binary codes are possible. Hence the range of decimal numbers has to be divided into 2^B steps and each step is represented by a binary code. Each step of decimal number is also called quantization step.

$$\therefore \text{ Quantization step size, } q = \frac{R}{2^B} = \frac{1-(-1)}{2^B} = \frac{2}{2^B} = \frac{1}{2^B - 2^{-1}}$$

$$= \frac{1}{2^{B-1}} = \frac{1}{2^b} = 2^{-b}$$

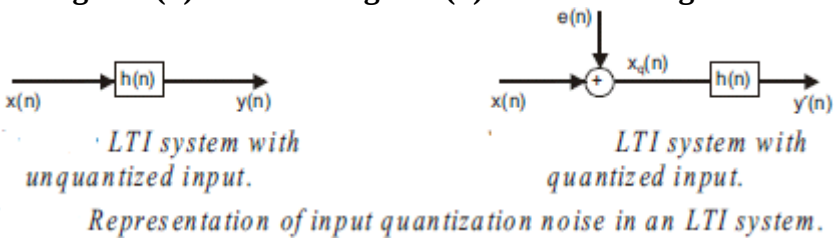
Where, R = Range of decimal number

B = Size of binary including sign bit

b = B – 1 = Size of binary excluding sign bit

Steady State Output Noise Variance (Power) Due to the Quantization Error Signal

The quantized input signal of a digital system can be represented as a sum of unquantized signal x(n) and error signal e(n) as shown in fig



Representation of input quantization noise in an LTI system.

Fig 4.3 Representation of input quantization noise in an LTI system.

In fig h(n) is the impulse response of the system and y(n) is the response or output of the system due to input and error signal. The response of the system is given by convolution of input and impulse response. For linear systems using distributive property of convolution the response y¢(n) can be written as shown in equation

$$y¢(n) = xq(n) * h(n)$$
$$= [x(n) + e(n)] * h(n)$$
$$= [x(n) * h(n)] + [e(n) * h(n)]$$

Let, y¢(n) = y(n) + e(n)

where, y(n) = x(n) * h(n) = Output due to input signal x(n). e(n) = e(n) * h(n) = Output due to error signal e(n).

The variance of the signal e(n) is called output noise power or steady state output noise power (or variance) due to the quantization error signal. Using autocorrelation function and the definition for variance of a discrete time signal, the expression for output noise power is

$$= \sigma_e^2 \sum_{i=1}^{N} \left[(z - p_i)\, H(z)\, H(z^{-1})\, z^{-1} \right]_{z = p_i}$$

where, p_i are poles of $H(z)\, H(z^{-1})\, z^{-1}$ only the poles that lie inside the unit circle in z-plane are considered.

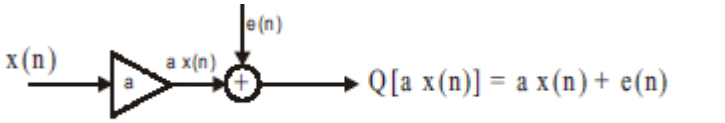
Product Quantization Error

In realization structures of IIR system, multipliers are used to multiply the signal by constants. The output of the multipliers i.e, the products are quantized to finite

word length in order to store them in registers and to be used in subsequent

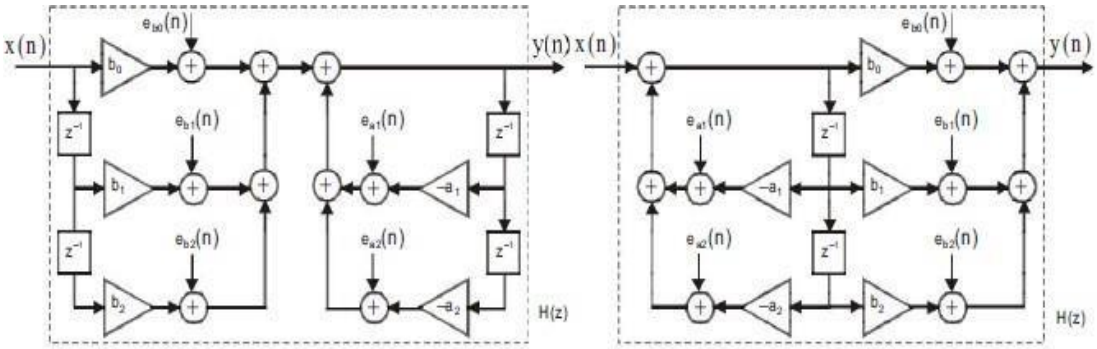
calculations. The error due to the quantization of the output of multiplier is referred to as product quantization error.

The Noise Transfer Function (NTF) is defined as transfer function from the noise source to the filter output (i.e., NTF is the transfer function obtained by treating the noise source as actual input).



Statistical model of fixed point product quantization.

Quantized product = Q[a x(n)] = a x(n) + e(n) where, a x(n) = Unquantized product
e(n) = Product quantization error signal



Second-order direct form-I. *Second-order direct form-II.*

Product quantization noise models of IIR systems for direct form realization.

Fig 4.4 Product quantization noise models of IIR systems for direct form realization

The total steady state noise variance at the output of the system due to product quantization errors is given by the sum of the output noise variances due to all the noise sources.

Output Noise Power (Roundoff Noise Power) Due to Product Quantization

Let, σ^2_{eTop} = Total output noise power due to product quantization error (or Total roundoff noise power)

$$\therefore \quad \sigma^2_{eTop} = \sigma^2_{e1op} + \sigma^2_{e2op} + \ldots + \sigma^2_{eMop}$$

Limit Cycles

During periodic oscillations, the output y(n) of a system will oscillate between a finite positive and negative value for increasing n or the output will become constant for increasing n. Such oscillations are called limit cycles. These oscillations are due to round-off errors in multiplication and overflow in addition.

Limit cycle oscillations are clearly unwanted (e.g. may be audible in speech/audio applications)

Limit cycle oscillations can only appear if the filter has feedback. Hence FIR filters cannot have limit cycle oscillations.

Types

1. **zero input limit cycles**
2. **Overflow limit cycles**

In recursive systems, if the system output enters a limit cycle, it will continue to remain in limit cycle even when the input is made zero. Hence these limit cycles are also called zero input limit cycles. In fixed point addition of two binary numbers the overflow occurs when the sum exceeds the finite word length of the register used to store the sum. The overflow in addition may lead to oscillations in the output which is referred to as overflow limit cycles

In a limit cycle the amplitudes of the output are confined to a range of values, which is called the dead band of the filter. For a first-order system described by the equation, y(n) = a y(n–1) + x(n), the dead band is given by,

$$\text{Dead band} = \pm\frac{2^{-B}}{1 - |a|} = -\frac{2^{-B}}{1 - |a|} \text{ to } +\frac{2^{-B}}{1 - |a|}$$

where, B = Number of binary bits (including sign bit) used to represent the product. For a second-order system described by the equation, y(n) = a1 y(n – 1) + a2 y(n – 2) + x(n), the dead band of the filter is given by,

$$\text{Dead band} = \pm\frac{2^{-B}}{1 - |a_2|} = -\frac{2^{-B}}{1 - |a_2|} \text{ to } +\frac{2^{-B}}{1 - |a_2|}$$

Scaling to Prevent Overflow

The two methods of preventing overflow are saturation arithmetic and scaling the input signal to the adder. In saturation arithmetic, undesirable signal distortion is introduced. In order to limit the signal distortion due to frequent overflows, the input signal to the adder can be scaled such that the overflow becomes a rare event.

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF ELECTRICAL AND ELECTRONICS

DEPARTMENT OF ELECTRONICS AND COMMMUNICATION ENGINEERING

UNIT - V
Multirate Signal Processing

V MULTIRATE SIGNAL PROCESSING

Introduction to Multirate signal processing

Single-rate systems: Sampling rates at the input and at the output and all internal nodes are the same.

Multirate systems: DSP systems with unequal sampling rates at various parts of the system.

The process of converting a signal from one sampling rate to another sampling rate is called sampling rate conversion.

There are two ways for sampling rate conversion in the digital domain. They are,

1. **Up-sampler / Up- Converter/ Interpolator**
2. **Down-sampler/ Decimator / Sub-sample**

Downsampling (or Decimation)

Down sampling or decimation is the process of reducing the sampling rate by an integer factor D.

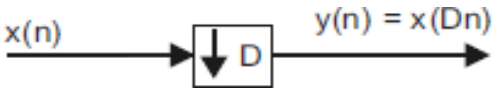


Fig.5.1 Decimator.

x(n) = Discrete time signal

D = Sampling rate reduction factor (and D is an integer)

Now, x(Dn) = Downsampled version of x(n)

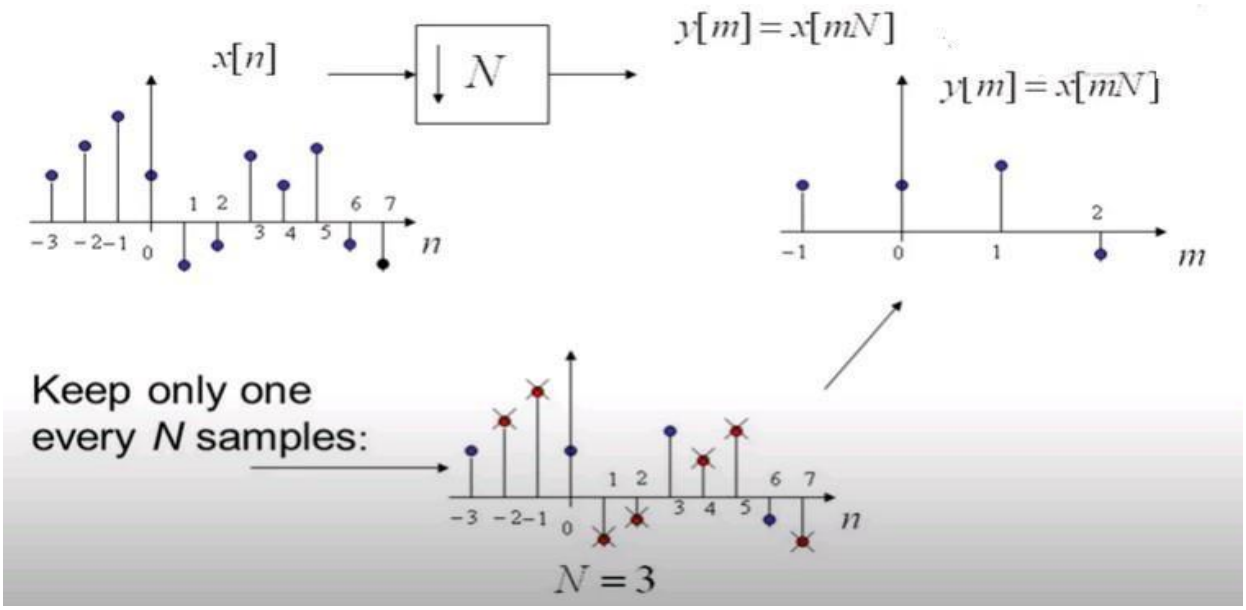


Fig 5.2 Time domain representation of decimation

Spectrum of Down sampler

The spectrum of Down sampler is given by

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

Anti-aliasing Filter

When the input signal to the decimator is not bandlimited then the spectrum of decimated signal has aliasing. In order to avoid aliasing the input signal should be bandlimited to p/D for decimation by a factor D. Hence the input signal is passed through a lowpass filter with a bandwidth of p/D before decimation. Since this lowpass filter is designed to avoid aliasing in the output spectrum of decimator, it is called anti-aliasing filter.

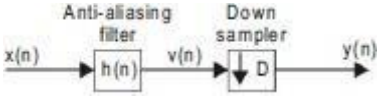


Fig 5.3 decimator with anti-aliasing filter.

Problem

sketch the spectrum of a down sampled signal for sampling rate reduction factor D = 2, 3 and 4.

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

$$= \frac{1}{2}X\left(e^{j\omega/2}\right) + \frac{1}{2}X\left(e^{j(\omega - 2\pi)/2}\right)$$

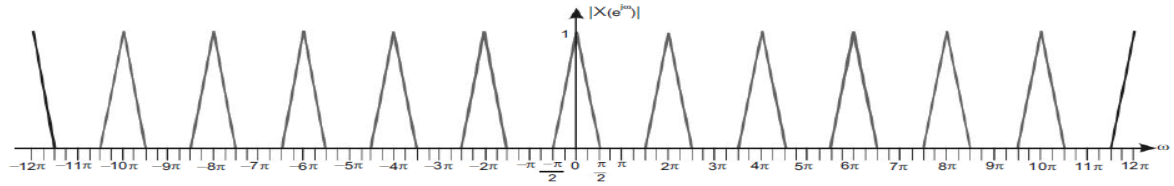


Fig 1.

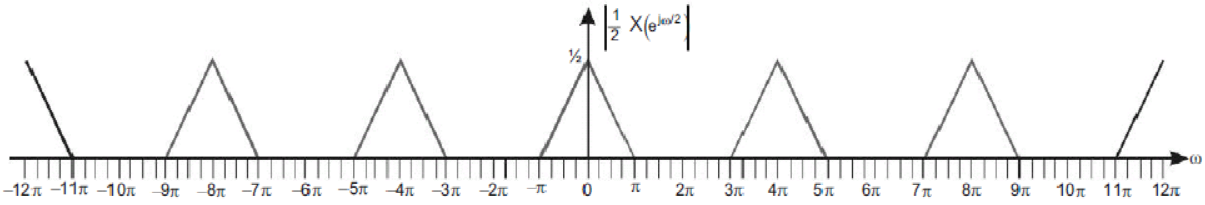


Fig 2.

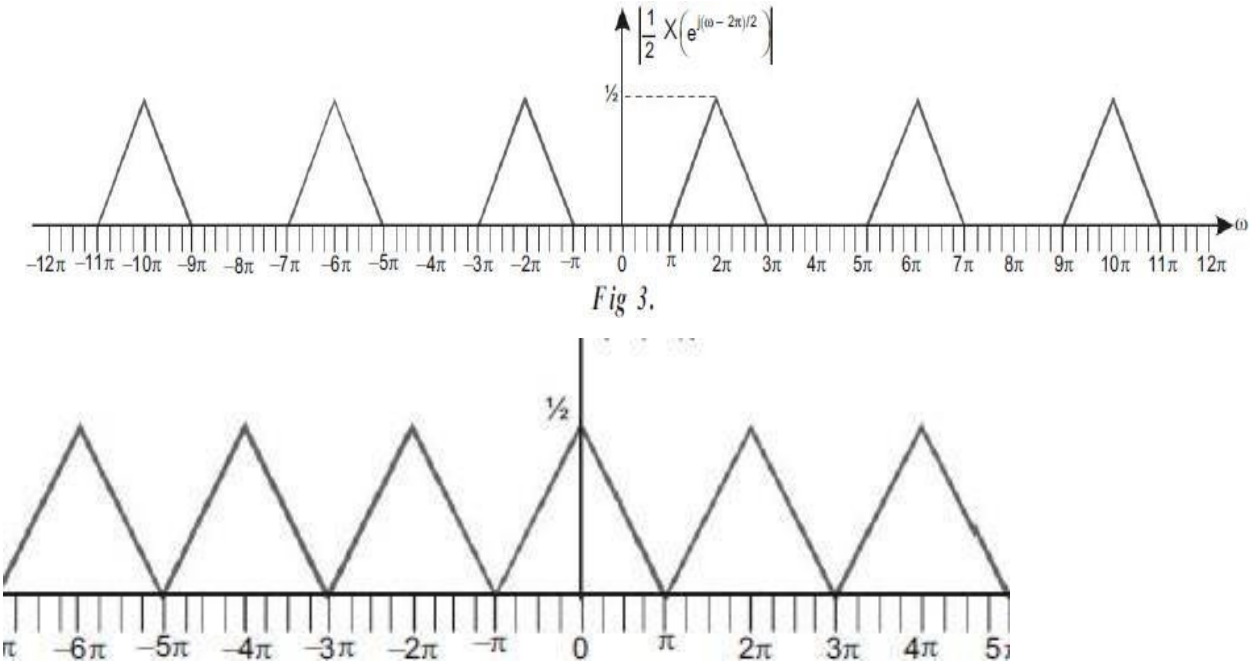


Fig. 5.4 spectrum of a down sampled signal for sampling rate reduction factor D = 2

Problem

Consider the discrete time signal shown in fig 1. Sketch the down sampled version of the signals for the sampling rate reduction factors, a) D = 2 b) D = 3.

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

Sampling rate reduction factor, D = 2.

$x(2n) = x_{D2}(n) = \{1, 1, 2, 2, 3, 3\}$

Sampling rate reduction factor, D = 3.

$x(3n) = x_{D3}(n) = \{1, -1, 2, -3\}$

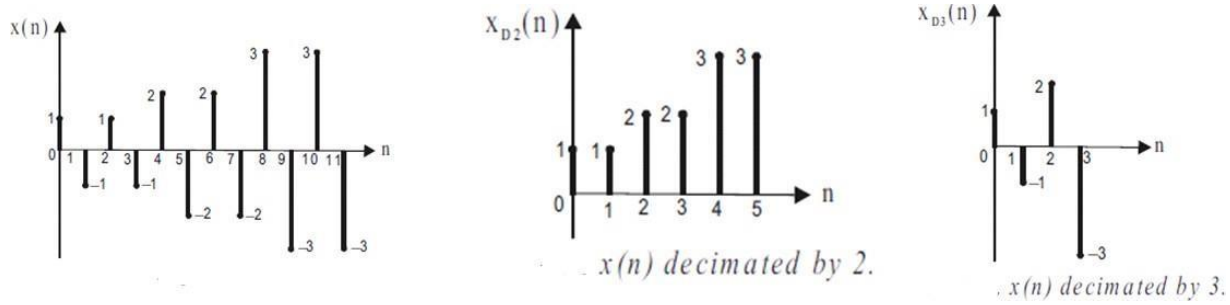


Fig 5.5 Down sampled version of the signals for the sampling rate reduction factors

D = 2 Upsampling (or Interpolation)

The upsampling (or interpolation) is the process of increasing the samples of the discrete

time signal.
Let, x(n) = Discrete time signal
I = Sampling rate multiplication factor (and I is an integer).
The device which perform the process of upsampling is called upsampler (or interpolator). Symbolically, the upsampler can be represented as shown in fig

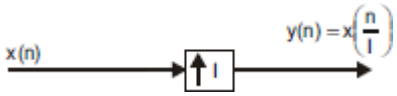


Fig 5.6 interpolator

Up sampling or interpolation is the process of increasing the sampling rate by an integer factor I.

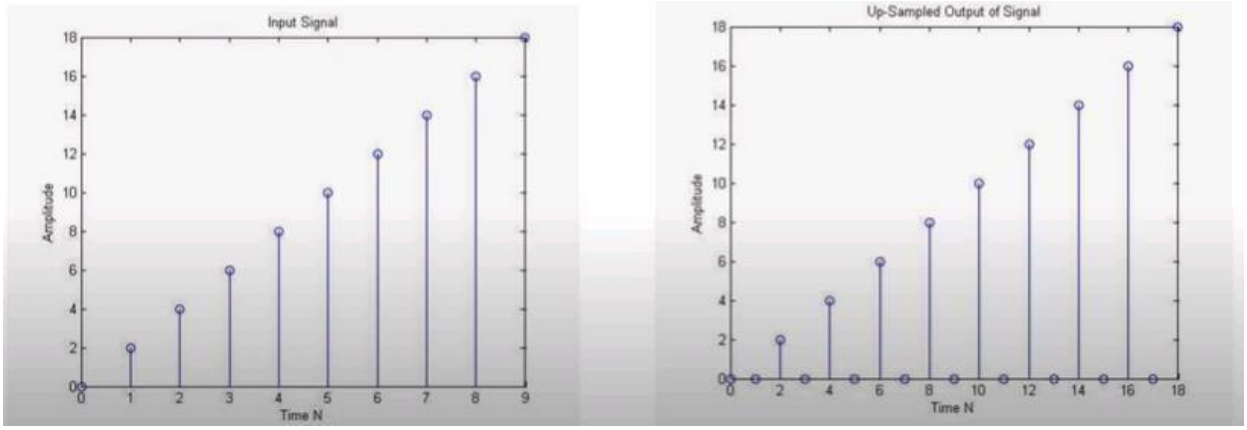


Fig 5.7 Time domain representation of interpolator

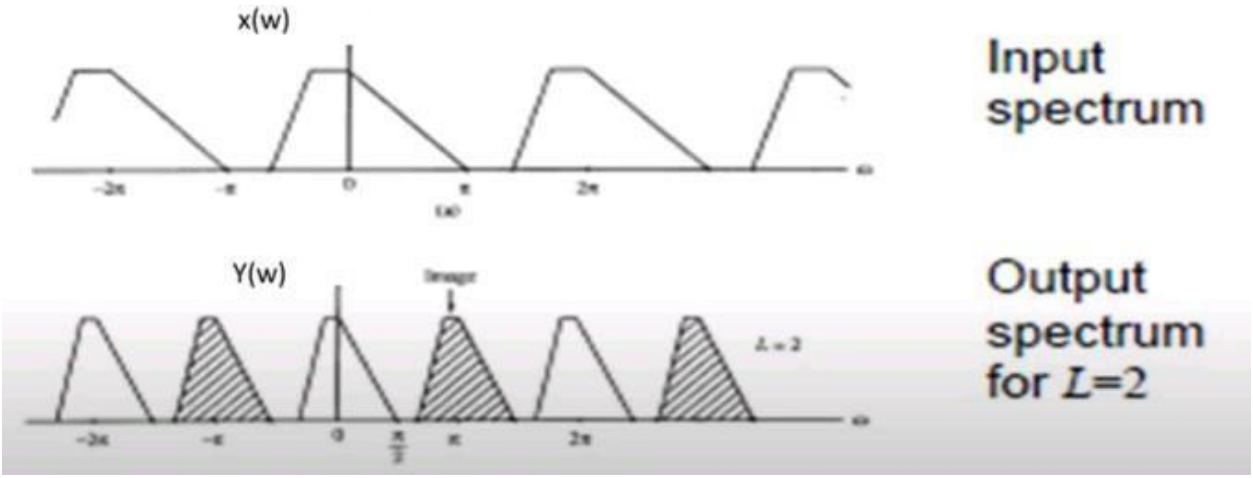


Fig 5.8 Spectrum of a upsampled signal for sampling rate reduction factor
L = 2 <u>Anti-imaging Filter</u>

The output spectrum of interpolator is compressed version of the input spectrum, therefore, the spectrum of upsampled signal has multiple images in a period of 2p. When

upsampled by a factor of I, the output spectrum will have I images in a period of 2p, with each image band limited to p/I. Since the frequency spectrum in the range 0 to π/I. are unique, we have to filter the other images. Hence the output of upsampler is passed through a lowpass filter with a bandwidth of π/I. Since this lowpass filter is designed to avoid multiple images in the output spectrum, it is called anti-imaging filter.

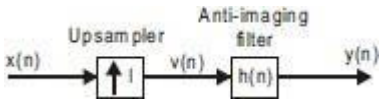


Fig 5.9. Interpolator with anti-imaging filter.

<u>Poly phase implementation of FIR filters for interpolator and decimator</u>

Potential computational savings can be made within the process of decimation, interpolation, and sampling-rate conversion. Polyphase filters is the name given to certain realisations of multirate filtering operations, which facilitate computational savings in both hardware and software.

<u>Polyphase Structure of Decimator</u>

In decimator, a lowpass filter called anti-aliasing filter is employed at the input in order to bandlimit the input signal, so that aliasing is avoided in the output spectrum of decimator. In order to reduce the computations in FIR filter, polyphase decomposition can be applied to FIR filter to decompose into L sub-filters.

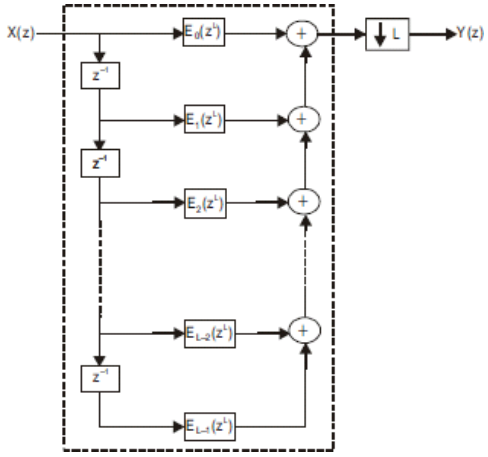


Fig 5.10 decimator with antialiasing filter

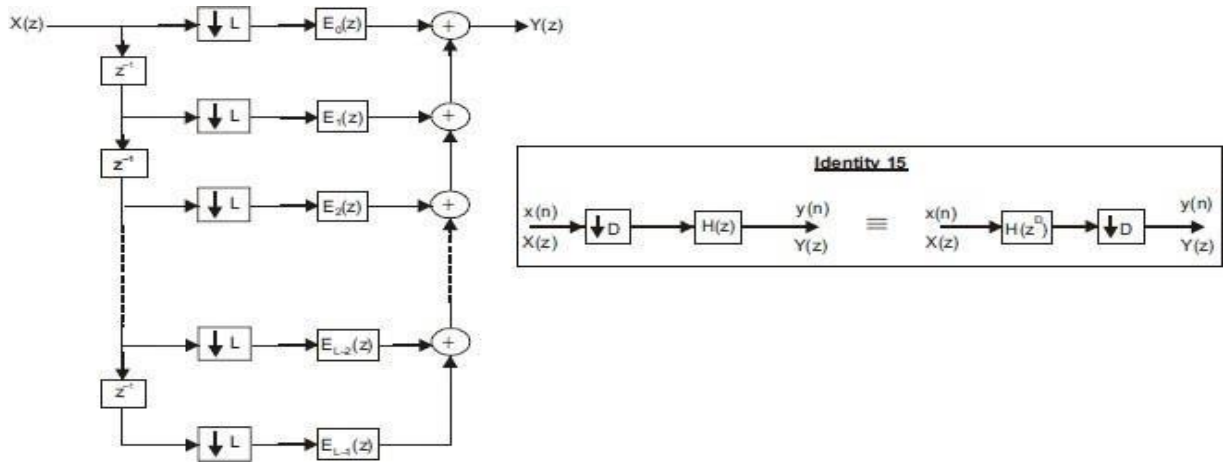


Fig 5.11 Decimator with antialiasing filter further deduction of fig 4.10 using identity.

Polyphase Structure of Interpolator

In interpolator, a lowpass filter called anti-imaging filter is employed at the output in order to eliminate the multiple images in the output spectrum of interpolator.

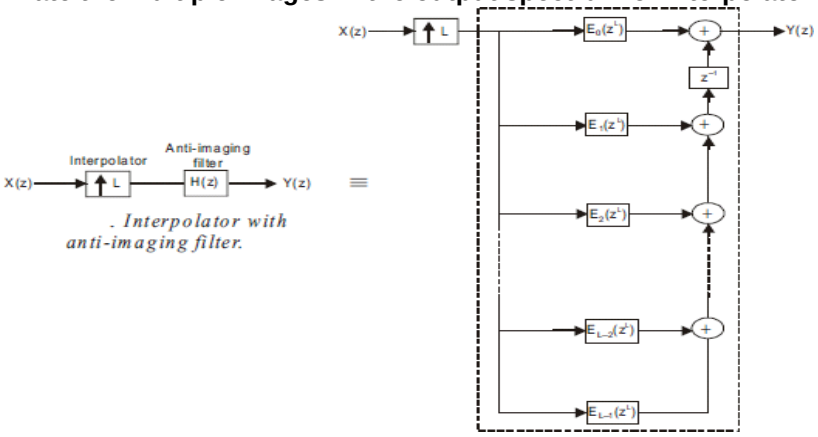


Fig 5.12 Interpolator with antialiasing filter

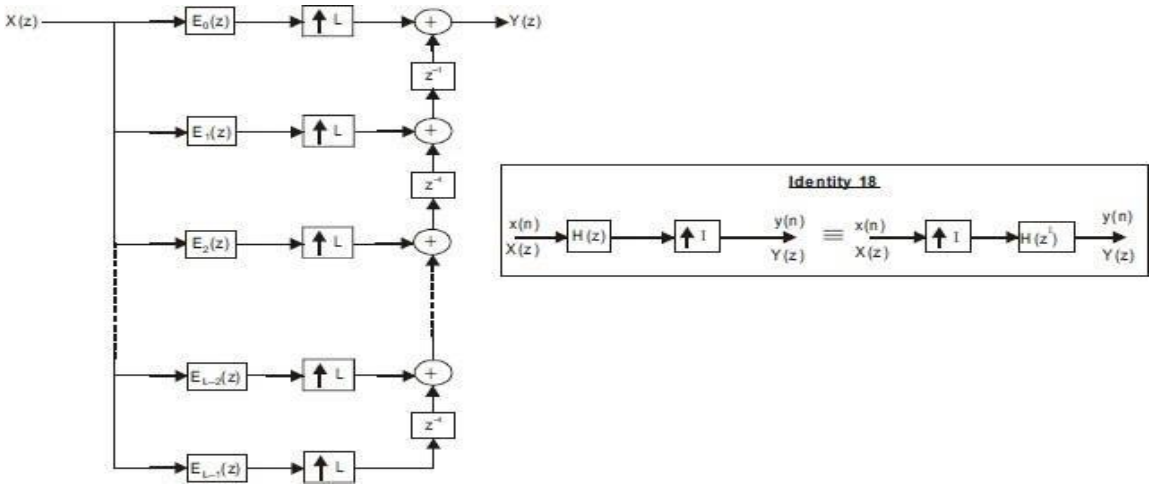


Fig 5.13 Interpolator with antialiasing filter further deduction of fig 4.12 using identity

Sampling rate conversion

A common use of multirate signal processing is for sampling-rate conversion. Suppose a digital signal x[n] is sampled at an interval T1, and we wish to obtain a signal y[n] sampled at an interval T2. Then the techniques of decimation and interpolation enable this operation, providing the ratio T1/T2 is a rational number i.e. L/M.
Sampling-rate conversion can be accomplished by L-fold expansion, followed by low-pass filtering and then M-fold
Decimation, It is important to emphasis that the interpolation should be performed first and decimation second, to preserve the desired spectral characteristics of x[n]. Furthermore by cascading the two in this manner, both of the filters can be combined into one single low-pass filter.

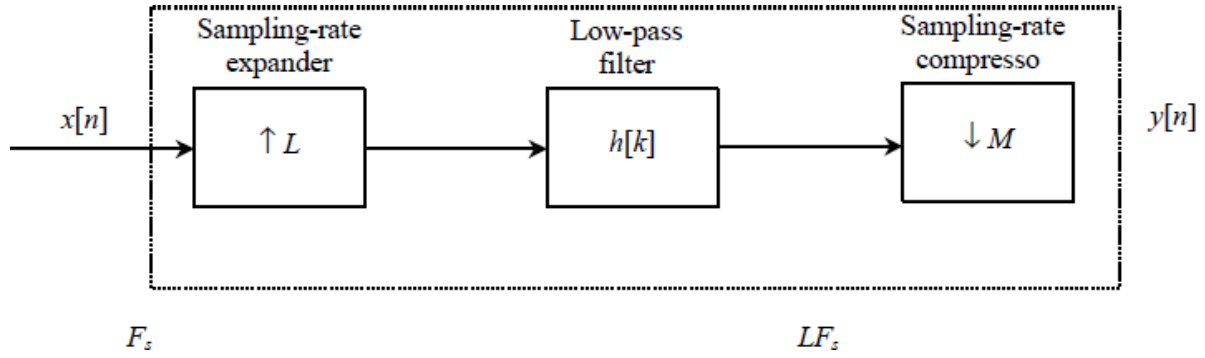


Fig 5.14.Sampling-rate conversion by expansion, filtering, and decimation

An example of sampling-rate conversion would take place when data from a CD is transferred onto a DAT. Here the sampling-rate is increased from 44.1 kHz to 48 kHz. To enable this process the non-integer factor has to be approximated by a rational number:

$$\frac{L}{M} = \frac{48}{44.1} = \frac{160}{147} = 1.08844$$

Design of narrow band filters

A common need in electronics and DSP is to isolate a narrow band of frequencies from a wider bandwidth signal. For example, you may want to eliminate 60 hertz interference in an instrumentation system, or isolate the signaling tones in a telephone network. Two types of frequency responses are available: the band-pass and the band-reject (also called a notch filter). Figure 5.15 shows the frequency response of these filters,

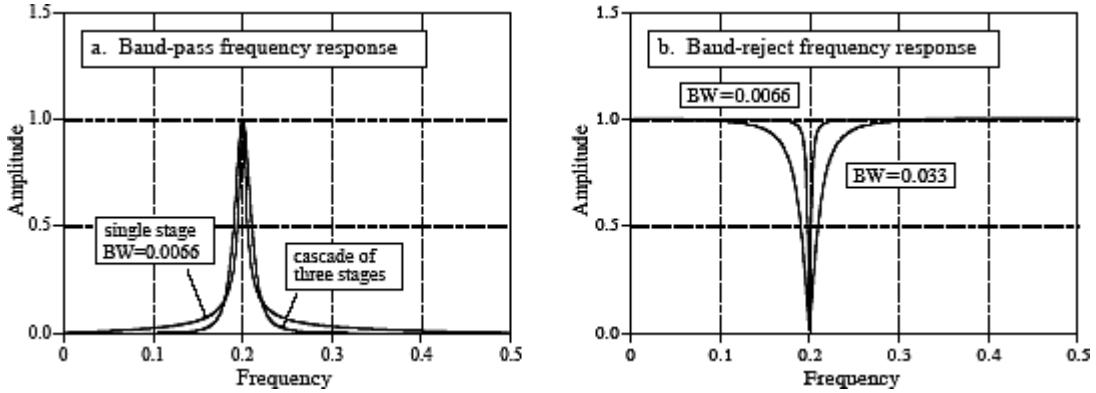


Figure 5.15 shows the frequency response of narrow band filters

Applications of Multirate signal processing

Applications of Multirate DSP Systems

Multirate signal processing is employed in the following systems.

1. **Sub-band coding of speech signals and image compression**
2. **QMF (Quadrature Mirror Filters) for realizing alias-free LTI multirate systems**
3. **Narrowband FIR and IIR filters for various applications**
4. **Digital transmultiplexers for converting TDM (Time Division Multiplexed) signals to FDM (Frequency Division Multiplexed) signals and vice versa**
5. **Oversampling A/D (Analog-to-Digital) and D/A (Digital-to-Analog) converters for high quality digital audio systems and data loggers (or digital storage systems)**
6. **In digital audio systems the sampling rates of broadcasted signal, CD (Compact Disc), MPEG (Motion Picture Expert Group) standard CD, etc., are different. Hence to access signals from all these devices, sampling rate converters are needed in digital audio systems.**
7. **In video broadcasting the American standard NTSC (National Television System Committee) and European standard PAL (Phase Alternating Line) employ different sampling rates. Hence to receive both the signals sampling rate converters are needed in video receivers.**

Advantages of Multirate Processing

The advantages of multirate processing of discrete time signals are given below.

1. **The reduction in number of computations**

2. **The reduction in memory requirement (or storage) for filter coefficients and intermediate results.**
3. **The reduction in the order of the system**
4. **The finite word length effects are reduced**

Digital Filter Banks

A digital filter bank is a set of bandpass filters. The digital filter banks can be classified into two types. They are,

i) Analysis filter banks

ii) Synthesis filter banks

Analysis Filter Banks

An analysis filter bank is a set of bandpass filters with common input. The analysis filter bank is used for spectrum analysis in which a signal is divided into a set of sub-band signals. The analysis filter bank consists of M numbers of sub-band filters so that the input signal x(n) is divided into M-numbers of sub-band signals.

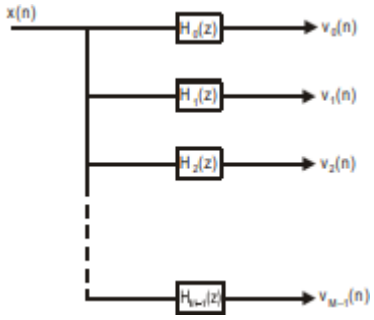


Figure 5.16 Analysis filter banks

Synthesis Filter Bank

A synthesis filter bank is a set of bandpass filters used to combine or synthesis a number of sub- band Signals into a single composite signal as shown in fig 9.31. The synthesis filter accepts M- numbers of sub-band signals $w_0(n)$, w1(n), w2(n), wM–1(n), combined to give a signal, y(n). In fact the synthesis filter bank perform the reverse process of analysis filter bank.

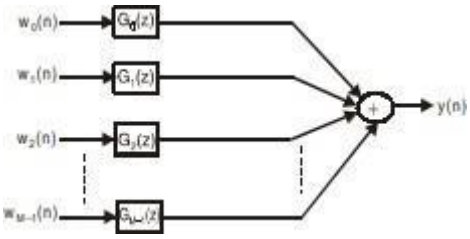


Figure 5.17 Synthesis Filter Bank

Applications of Multirate signal processing

In the digital audio industry, it is a common requirement to change the sampling rates of band-limited sequences. This arises for example when an analog music waveform x,(t) is to be digitized. Assuming that the significant information is in the band 22 kHz a minimum sampling rate of 44 kHz is suggested. It is, however, necessary to perform analog filtering before sampling to eliminate aliasing of out-of-band noise. Now the requirements on the analog filter it should have a fairly flat passband and a narrow transition band (so that only a small amount of unwanted energy is let in). Optimal filters for this purpose (such as elliptic filters, which are optimal in the minimax sense) have a very nonlinear phase response around the bandedge (i.e., around 22 kHz). In highquality music this is considered to be objectionable. A common strategy to solve this problem is to oversample x,(t) by a factor of two (and often four). Further applications of muItirate fiIter banks in digital audio are Subband Coding of Speech and Image Signals.

Sub-band Coding of Speech Signals

In sub-band coding of speech signals, the speech signal is divided into sub-bands, decimated, encoded and transmitted to the receiver system. On the receiver side the subband signals are decoded, interpolated and synthesized into the original speech signal. The figure below shows the subband coding of speech signal.

In the transmission side, the input signal is split into M-numbers of non-overlapping frequency bands using an analysis filter bank consisting of M-numbers of bandpass filters. The output of each bandpass filter is decimated by a factor of D. The output of decimators

are encoded and transmitted. On the reception side, the received sub-band signals are decoded and then interpolated to recover the missing samples. The output of interpolators are applied to a synthesis filter bank consisting of M-numbers of bandpass filters to recover the original signal.

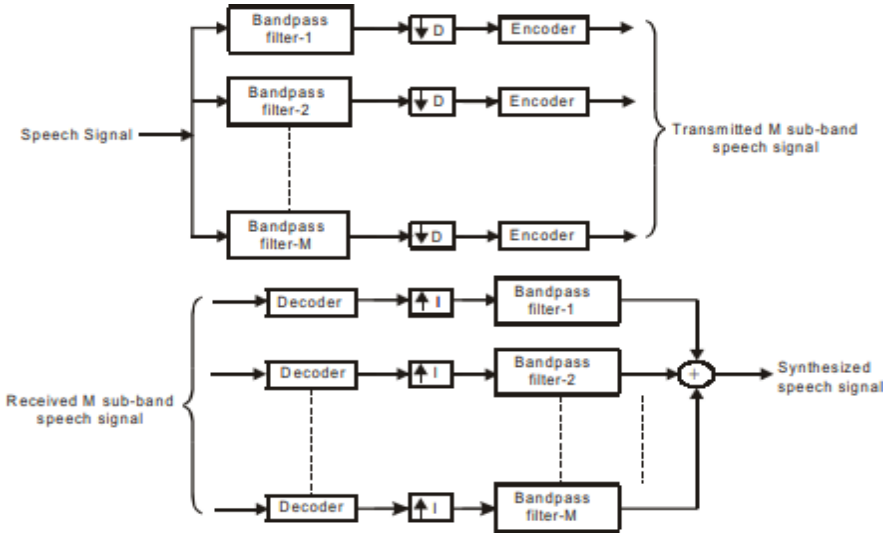


Figure 5.18 Sub-bands Coding of Speech Signals.

Speech compression

The processing of speech involves the analysis, coding, decoding, and synthesis of speech sounds. The speech analyzer consists of normalizers, syllable, segmenters, sound recognizers, sequencers, adapters, and memories which convert the speech elements into a code. The speech synthesizer converts the code to speech by reproducing prerecorded speech elements. There are many applications for the speech analyzer and synthesizer ranging from limited vocabulary to complete communication systems. The most important systems for the communication of speech information are the telephone, phonograph, radio, sound motion picture, and television.

The main objective in the analysis of speech as applied to communication systems is to provide a savings in the channel capacity required for transmission.• There are several considerations involved in the use of the different speech elements in communication systems as follows: the bit rate for the transmission of speech, the segmentation of speech,

the analysis of speech, the synthesis of speech. In order to analyze the different types of speech, there must be some means for the segmentation of the flow of speech. The segmentation involves sentences, word , syllables and phonemes.

Segmentation of speech into syllables reduces the number of speech segments and Reduction of bandwidth. In conventional speech processing applications, speech signal is encoded using fixed number of bits over the entire speech signal band. During the process, the bandwidth requirement for speech transmission is relatively high which is of concern. The QMF (Quadrature Mirror Filter) banks are the fundamental building blocks for spectral splitting. The aim is to design a QMF filter and then pass a speech signal through it. In speech signals most of the energy is present in the lower frequency bands. Signal coding is the act of transforming the signal at hand to a more compact form, which can then be transmitted with considerably smaller memory. The motivation behind this is the fact that access to the unlimited amount of bandwidth, which is not possible.

Therefore there is a need to code and compress speech signals. By taking advantage of the fact that most of the energy is present in a particular frequency band we can split the signal into various bands depending on the information content and then code the subband signals separately. The basic theory of multirate digital signal processing is introduced in this section along with the two Sampling rate alteration devices namely up-sampler and down-sampler.

<u>Elimination of interference:</u>

Multirate digital signal processing has a very important role in sub band coding of speech, audio ,video and multiple carrier data transmission because of the high computational efficiency of the multirate algorithms. The performance of a filter bank based interference detection and suppression method to extract the original speech from the interference contaminated speech using the perfect reconstruction (PR) property of the Cosine Modulated filter bank.

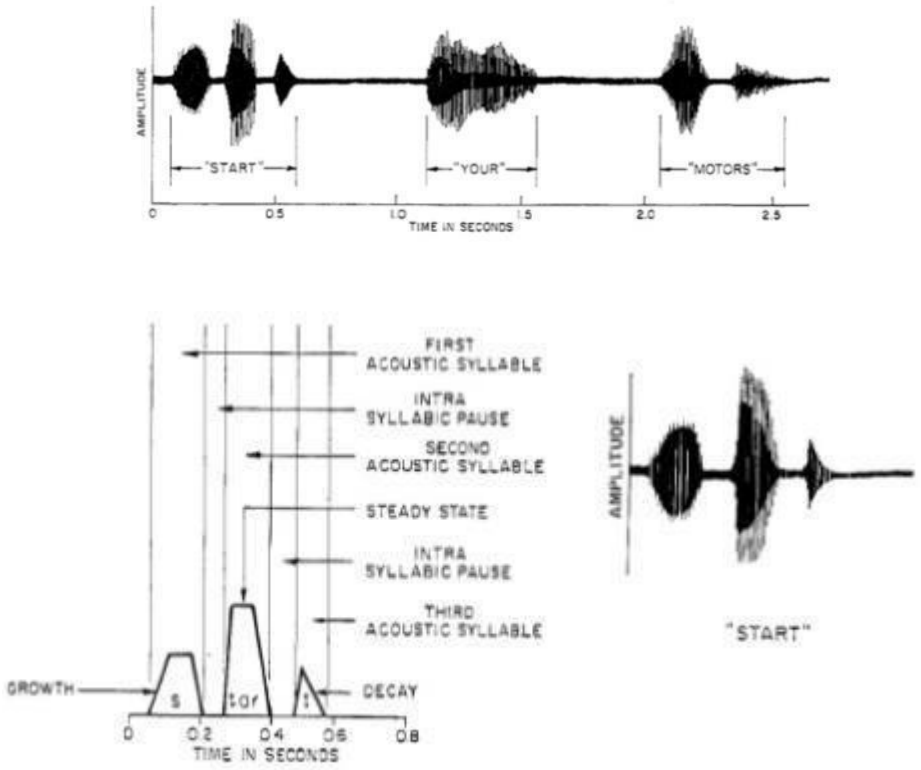


Figure 5.18 Segmentation of speech into syllables

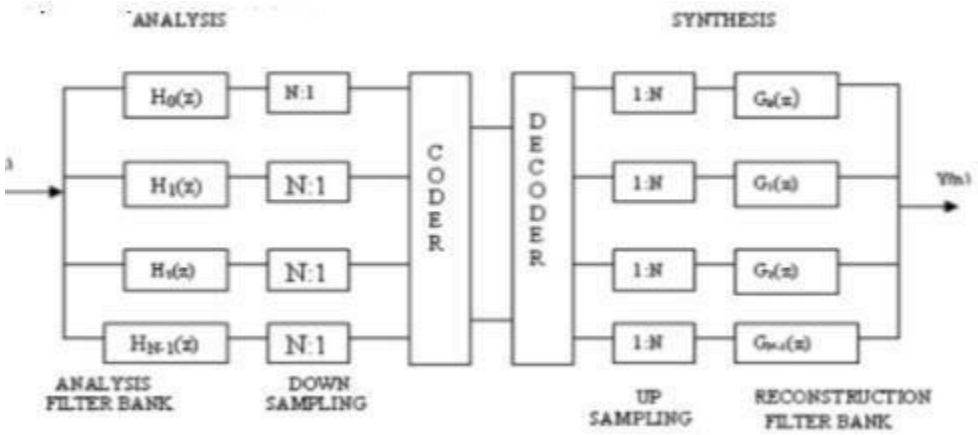


Figure 5.19 QMF filter

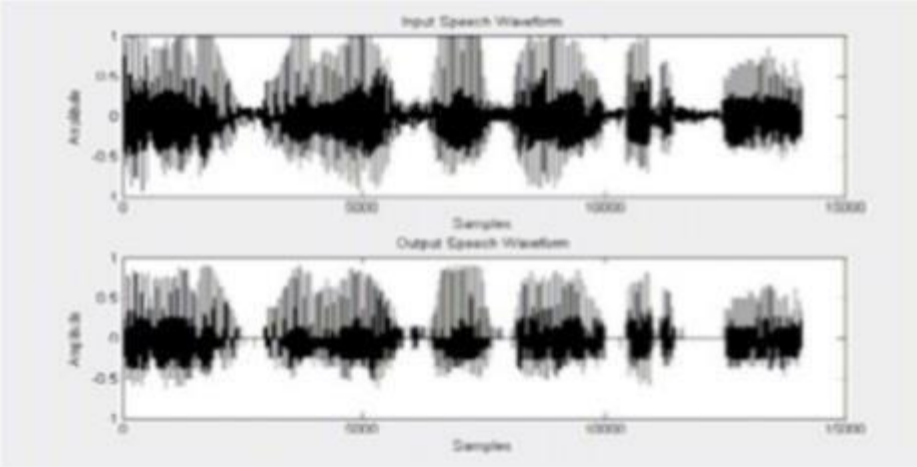


Figure 5.20 Sampled output of speech signal.

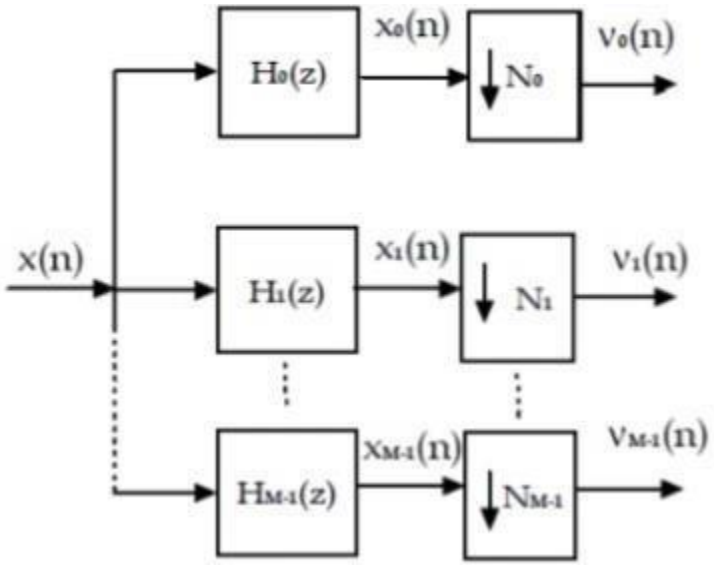


Figure 5.21 Cosine modulated filter bank.

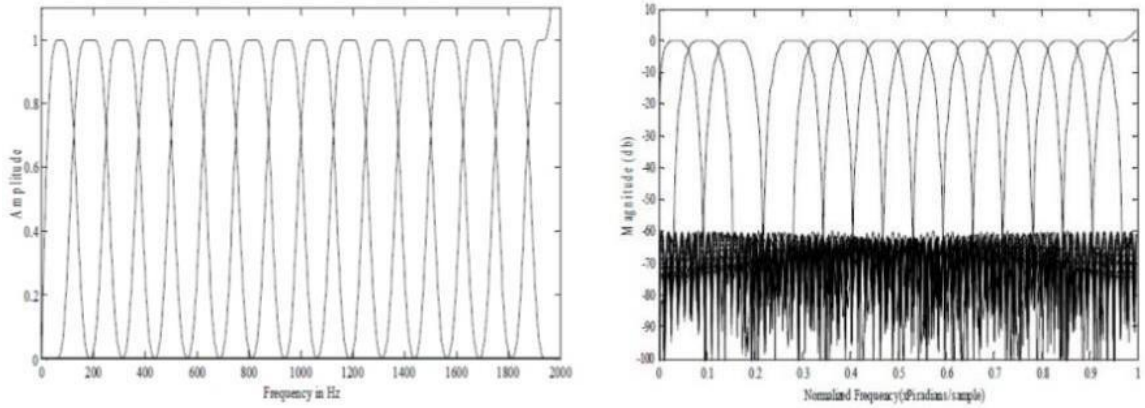


Figure 5.22 Signal with added interference

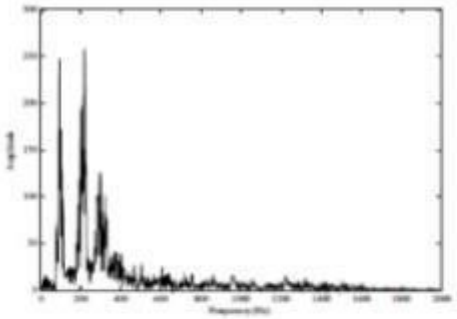


Figure Spectrum of the original signal

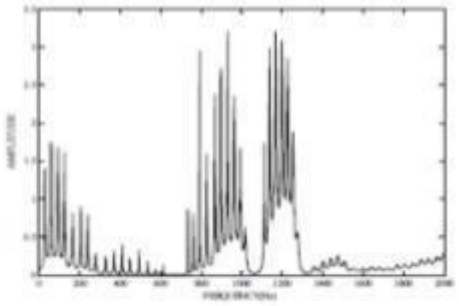


Figure Spectrum of the interference added signal

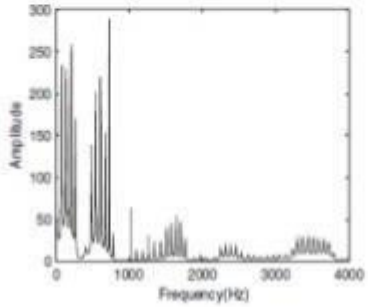


Figure The spectrum of the signal with interference suppression resembling the original spectrum

Figure 5.23 Simulated response of speech process.

The interference suppressor is a critically sampled filter bank system. Modulated filter banks are used to form analysis-synthesis filter banks that divide the received signal into several channels (analysis part), and reconstruct the original signal from the sub-channels (synthesis part). When a signal with added interference is applied to the analysis filter banks, the signal interference appears at the output of one of the filter banks. The spectrum of each sub band signal is estimated to identify the interference bands. For interference suppression, the sub channels affected by the interference are not included in the synthesis filter bank, resulting in notch filtering

<u>Adaptive filter</u>

The goal of adaptive filters are to maintain or derive desired output signal characteristics from a FIR or IIR filter. This goal is obtained via a feedback loop structure that feeds measure of undesired signal characteristics (error) to the filter under consideration and subsequently the filter updates its filter kernel with the fed coefficients to generate or maintain the desired output signal characteristics. The calculation of new coefficients based on the error signal feedback which is to be minimized is powered by some adapting

algorithms. The error is defined as the deviation of output signal from the desired signal characteristics, such that, where d(n) is the desired signal, y(n) is the output signal and e(n) is the error signal, then the following formulas holds.

$$y(n) = \sum_{i=0}^{N-1} W_i(n)\, x(n-i)$$

To derive the desired signal from the system, we first have to measure the error signal through finding out mathematical correlation between samples of output signal and desired signal. In short, from a higher point of view, this error signal is measured by subtracting the first signal from the latter signal. Then, this error signal is optimally minimized via updating operating filter's coefficients through a live feedback loop.

The use of adaptive filters can be divided majorly into two groups. Firstly, to continuously maintain the output signal unchanged from a running filter. Secondly, to approximate a desired signal from the output signal of a filter. These both approach use the same fundamental structure of the adaptive filter but they varies in terms of orientation and applications.

Adaptive filters can be mainly structurally realized into two ways, namely, spatially and functionally. Spatial structure discusses about the organization of filter components without restricting corresponding filters desired functional output. On the other hand, functional structure discusses about the functional role of the sub-systems of each adaptive filter.

Spatial Structure or Block Diagram

The most common used structure are direct form, cascade form, parallel form and lattice. Transversal layout of adaptive filters are most commonly used, however, lattice layout is also used when its advantages overrides the advantages of transversal layout.

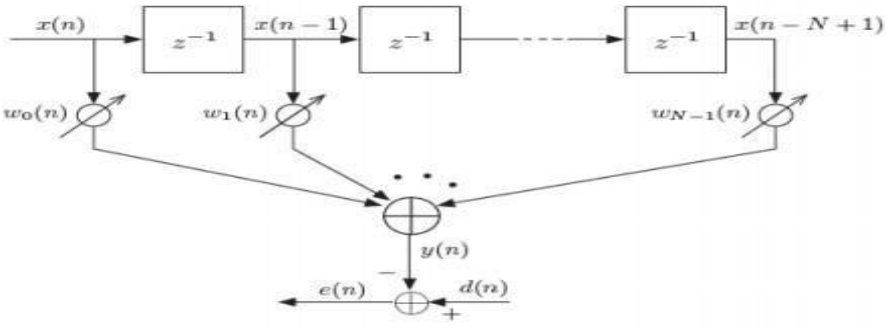


Figure 5.24. Spatial Structure or Block Diagram

Error signal is the difference between output signal and desired signal. That is to say that, error signal is the amount of signal component that adaptive filter optimally removes when it converges and thus arriving at the desired condition.

Adaptive control algorithm is the algorithm that adaptive filter uses to iteratively calculate the new coefficients that optimally reduces the power of error signal. The choice of adaptive control algorithm depends on the data class, memory resources, computational time, energy requirements and overall cost. The L-MSE and LSE are two commonly used algorithm to calculate the updated coefficients.

<u>Musical sound processing:</u>

The musical sound generated by a musical instrument is due to mechanical vibrations produced by a primary oscillator and then making other parts of the instrument to vibrate. For example, in a violin the primary oscillator is a stretched piece of string and it is vibrated by drawing a bow across it, which in turn vibrates the wooden body of the violin, and these vibrations make the surrounding air to vibrate, which produces the musical sound.

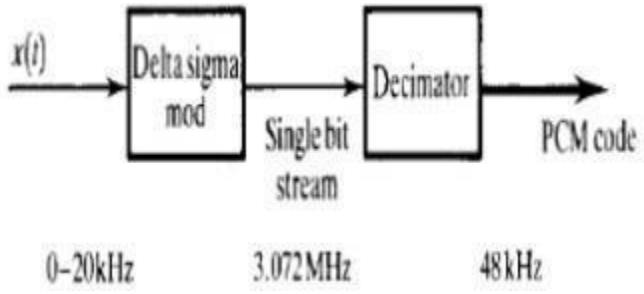


Figure 5.25. High quality Analog to Digital conversion for digital audio

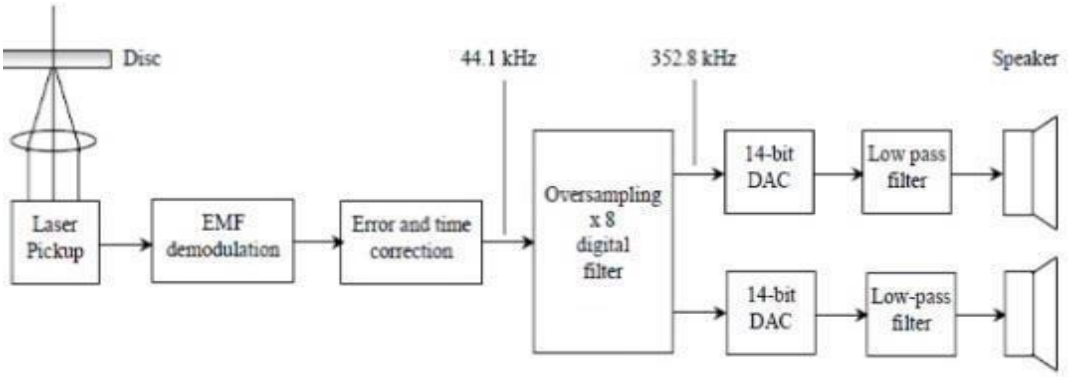


Figure 5.26. Multirate systems are used in a CD player when the music signal is converted from digital into analog

<u>Digital Music Synthesis:</u> Music synthesis plays an important role in multimedia applications, modern entertainment, and professional music systems. The various music synthesis techniques used in the commercial systems are wavetable synthesis, spectral modeling synthesis, nonlinear synthesis (or FM synthesis) and physical modeling synthesis. In wavetable synthesis method, the digital data of one period of the desired musical tone is stored in a table called wavetable. Then, using an IIR filter with no input and the stored data as initial condition, the musical signal is constructed whenever needed. In spectral modeling synthesis, the mathematical equation representing the sound signal is used to generate the required music. The musical sound can be represented by an equation consisting of summation of sinusoidal signals. A musical tone consists of a fundamental tone frequency and its harmonics. Using suitable signal generation algorithm, thedesired

musical tone can be generated. In nonlinear synthesis, the musical sound signal is represented as a nonlinear frequency modulated sinusoidal signal containing a fundamental frequency and harmonics of modulating signal. Using signal generation algorithm, various musical tones can be generated for various fundamental frequency. This method cannot be used to generate musics of natural instruments. In physical modeling synthesis, a model of musical instrument like transfer function is constructed and the system model is implemented in a digital hardware, that can be used to generate the musics of an instrument.

The recording of musical programs are generally made in an acoustically inert studio. The sound of each instrument is separately recorded using microphones placed closed to it and then they are mixed using mixing system by a sound engineer. During mixing phase, various audio effects are artificially generated using signal processing circuits and devices. The modern trend is to use digital signal processing for these applications. Some of the special effects that can be implemented during mixing process are echo generation, reverberation, and chorus generation. Also, the musical sound signals can be passed through equalizers to provide amplification or attenuation of some of the tone frequencies

<u>Image enhancement.</u>

The Mach band phenomenon is a good example of this property of the HVS. In an Image 21 consisting of adjacent rectangular bands of different gray levels (called Mach band), the perceived gray level near the edges is different than in the middle of the rectangles. The edge near the darker band appears lighter and the one near the lighter band appears darker than the middle of the rectangle.

For 2-D signals (images) only, however the concepts can be extended to M-D signals. A 2-D analog signal x a(t) is a function of the variable t which can be defined as a column vector

$$\bar{t} = \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \quad T_1 = \begin{bmatrix} T_{11} \\ T_{21} \end{bmatrix} \quad T_2 = \begin{bmatrix} T_{12} \\ T_{22} \end{bmatrix} \quad \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}$$

D is the sampling matrix made up of sampling vectors T1 and T2

The matrix D that generates LAT(D) is not unique and the lattice may or may not be separable. A separable lattice is a lattice that can be represented by a diagonal matrix. For example, the rectangular lattice has a sampling matrix form of Dr and matrix Dh can generate a hexagonal sampling lattice.

$$D_r = \begin{bmatrix} T_{11} & 0 \\ 0 & T_{22} \end{bmatrix} \quad D_h = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & -T_{22} \end{bmatrix}$$

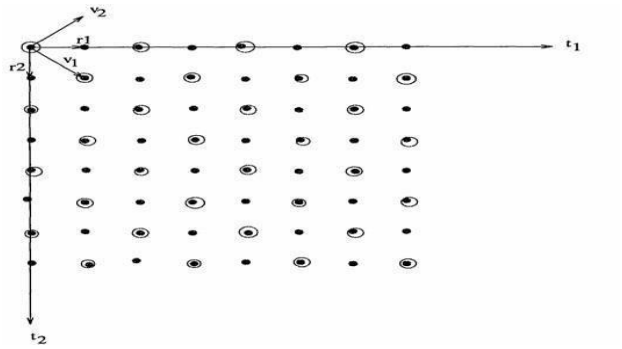


Figure 5.27 Hexagonal resampling and decimation by 2 of a rectangular grid

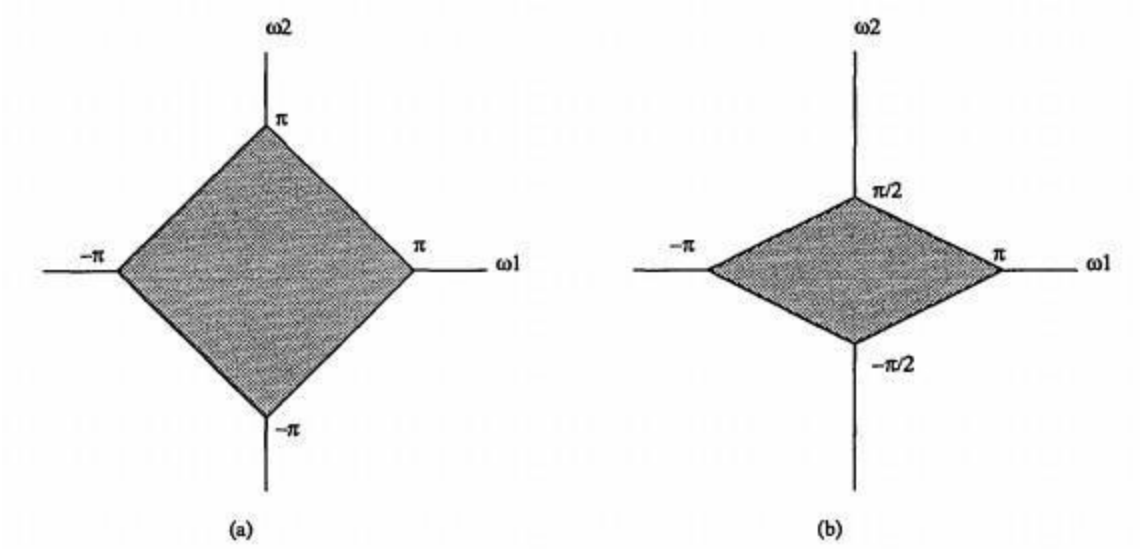


Figure 5.28 Frequency domain support for hexagonal decimation filters: (a) Hex decimation by 2 (b) Hex decimation by 4

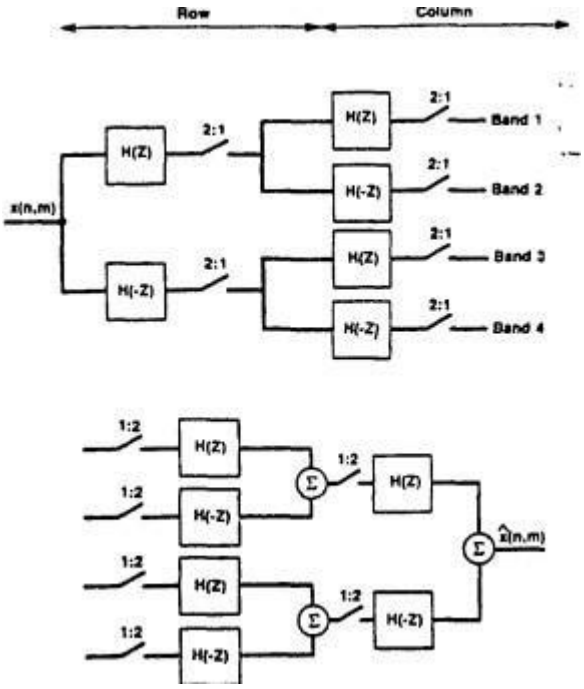


Figure 5.29 Two-dimensional separable QMF bank system.

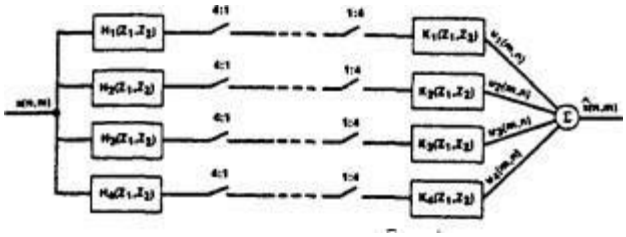
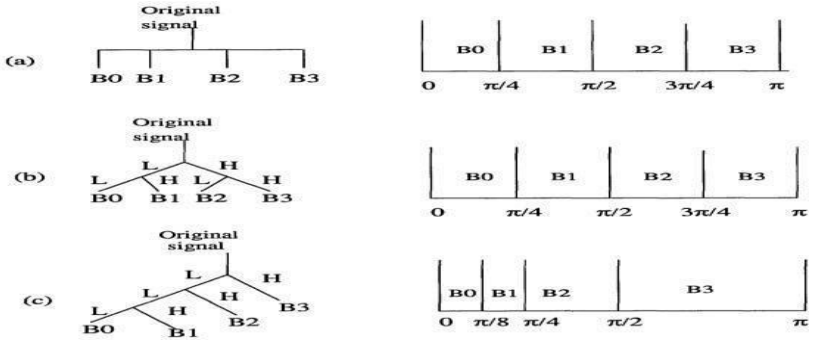


Figure 5.30. Two-dimensional non-separable filter bank system



L stands for lowpass branch and H stands for highpass branch Figure 5.31. 1-D Subband decomposition structures.