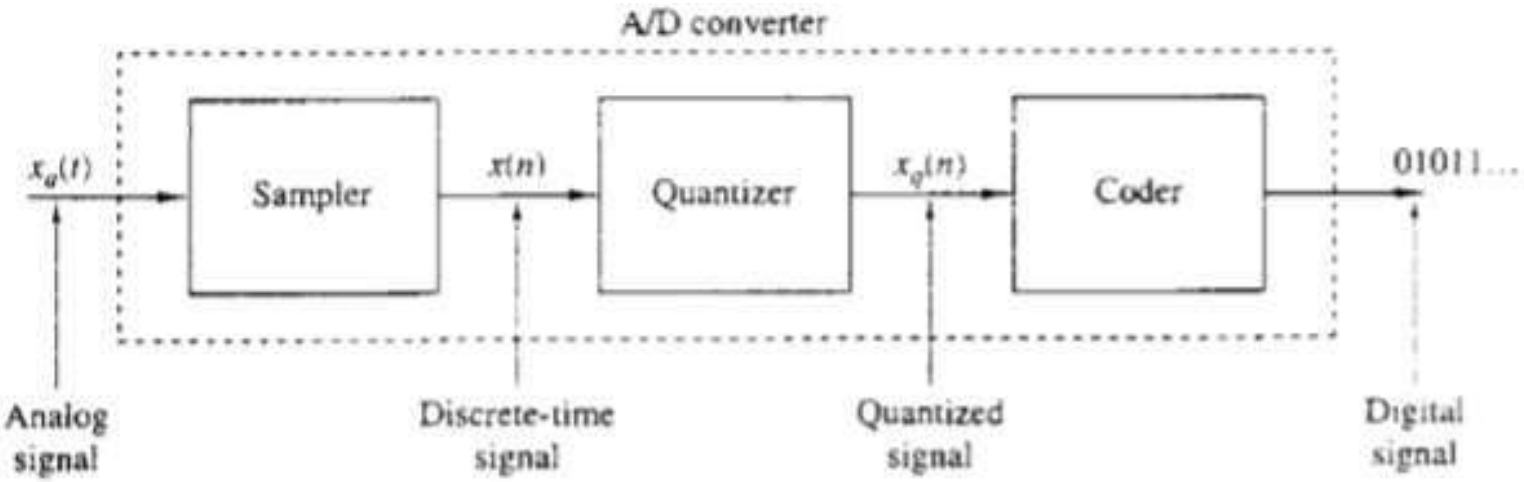
SIGNAL CONVERSION SYSTEMS

Analog-to-Digital and Digital-to-Analog conversion

- Most signals of practical interest, such as speech, biological signals, seismic signals, audio and video signals are analog.
- To process analog signals by digital means, it is first necessary to convert them into digital form i.e., as a sequence of numbers having finite precision.
- This procedure is called analog-to-digital (A/D) conversion.

A/D conversion viewed as a three-step process



1. Sampling. Conversion of a continuous-time signal into a discrete-time signal obtained by taking "samples'" of the continuous-time signal at discrete-time instants. If $x_a(t)$ is the input to the sampler, the output is $x_a(nT)$, where T is called the sampling interval.
2. Quantization. Conversion of a discrete-time continuous-valued signal into a discrete-time, discrete-valued (digital) signal, $x_q(n)$. The value of each signal sample is represented by a value selected from a finite set of possible values.
3. Coding. In the coding process, each discrete value $x_q(n)$ is represented by a b-bit binary sequence.

Digital-to-Analog conversion

- In many cases of practical interest (e.g., speech processing) it is desirable to convert the processed digital signals into analog form. (Obviously, we cannot listen to the sequence of samples representing a speech signal).
- The process of converting a digital signal into an analog signal is known as digital-to-analog (D/A) conversion.
- A simple form of D/A conversion is called a zero-order hold or a staircase approximation.
- Other approximations are possible, such as linearly connecting a pair of successive samples (linear interpolation), fitting a quadratic through three successive samples (quadratic interpolation), and so on.
- For signals having a limited frequency content (finite bandwidth), the sampling theorem specifies the optimum form of interpolation.

Sampling and Aliasing

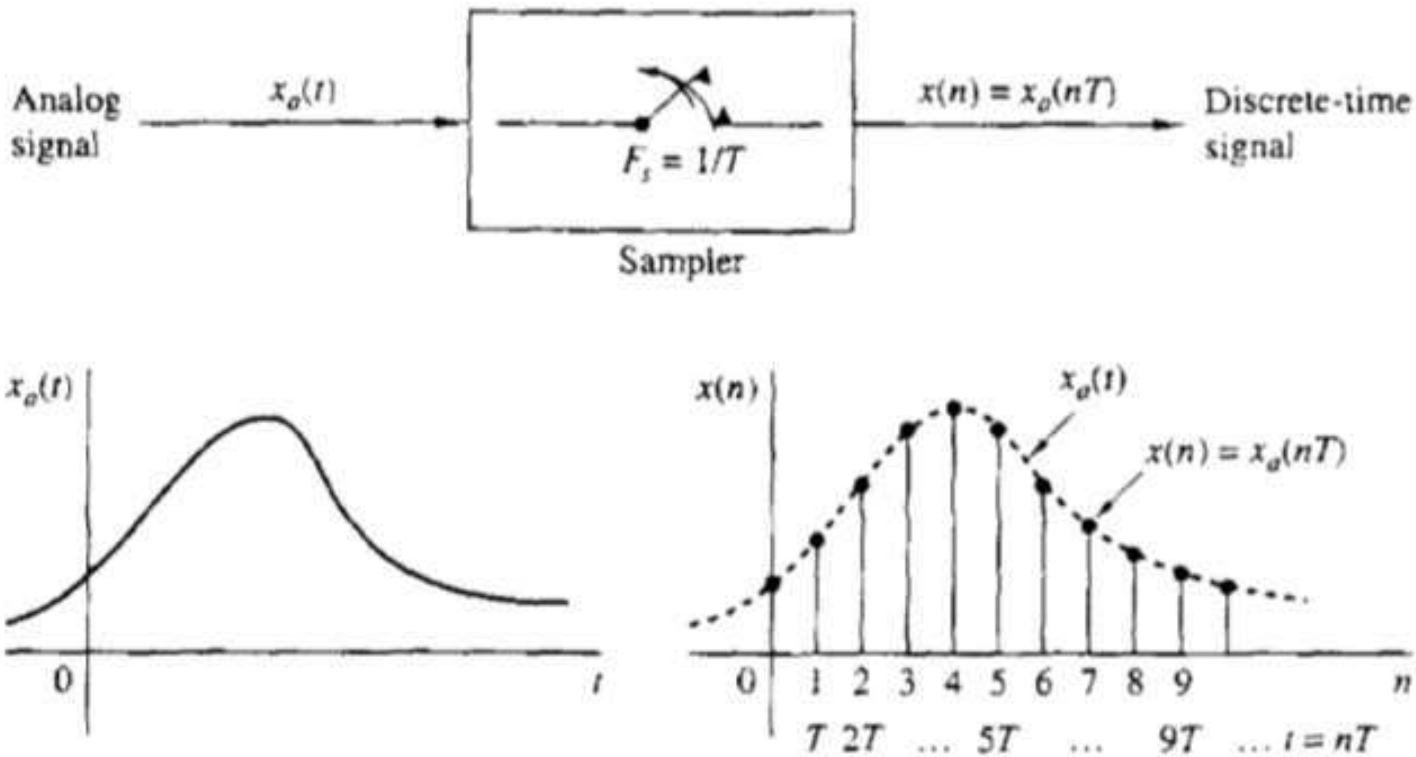
- Sampling does not result in a loss of information, nor does it introduce distortion in the signal if the signal bandwidth is finite.
- In principle, the analog signal can be reconstructed from the samples, provided that the sampling rate is sufficiently high to avoid the problem commonly called aliasing.
- On the other hand, quantization is a noninvertible or irreversible process that results in signal distortion.
- The amount of distortion is dependent on the accuracy, as measured by the number of bits, in the A/D conversion process.
- The factors affecting the choice of the desired accuracy of the A/D converter are cost and sampling rate. In general, the cost increases with an increase in accuracy and/or sampling rate.

Sampling of Analog Signals

- There are many ways to sample an analog signal. Periodic or uniform sampling is the type of sampling used most often in practice.
- This is described by the relation

$$x(n) = x_a(nT), \qquad -\infty < n < \infty$$

where $x(n)$ is the discrete-time signal obtained by "taking samples" of the analog signal $x_a(t)$ every T seconds.



- The time interval *T* between successive samples is called the sampling period or sample interval.
- Reciprocal 1/ $T = F_s$ is called the sampling rate (samples per second) or the sampling frequency (hertz).
- Periodic sampling establishes a relationship between the time variables *t* and *n* of continuous-time and discrete-time signals, respectively. Indeed, these variables are linearly related through the sampling period *T* or, equivalently, through the sampling rate F_s, as

$$t = nT = \frac{n}{F_s}$$

- As a consequence, there exists a relationship between the frequency variable *F* (or Ω) for analog signals and the frequency variable *f* (or ω) for discrete-time signals.

Consider an analog sinusoidal signal of the form $x_a(t) = A\cos(2\pi Ft + \theta)$

which, when sampled periodically at a rate *Fs* samples per second, yields

$$x_a(nT) \equiv x(n) = A\cos(2\pi FnT + \theta)$$
$$= A\cos\left(\frac{2\pi nF}{F_s} + \theta\right)$$

If we compare $x_a(nT)$ with

$$x(n) = A\cos(2\pi fn + \theta), \quad -\infty < n < \infty$$

we note that the frequency variables F and *f* are linearly related as,

$$f = \frac{F}{F_s}$$

or, equivalently, as

$$\omega = \Omega T$$

The range of the frequency variable F or Ω for continuous-time sinusoids are

$$-\infty < F < \infty$$
$$-\infty < \Omega < \infty$$

However, the situation is different for discrete-time sinusoids.

$$-\frac{1}{2} < f < \frac{1}{2}$$

$$-\pi < \omega < \pi$$

By substituting $f=F/F_s$ and $\omega = \Omega T$ into the above representation, we find that the frequency of the continuous-time sinusoid when sampled at a rate Fs = $1/T$ must fall in the range

$$-\frac{1}{2T} = -\frac{F_s}{2} \le F \le \frac{F_s}{2} = \frac{1}{2T}$$

or, equivalently.

$$-\frac{\pi}{T} = -\pi F_s \le \Omega \le \pi F_s = \frac{\pi}{T}$$

From these relations we observe that the fundamental difference between continuous-time and discrete-time signals is in their range of values of the frequency variables F and f, or Ω and ω. Periodic sampling of a continuous-time signal implies a mapping of the infinite frequency range for the variable F (or Ω) into a finite frequency range for the variable f (or ω).

- Since the highest frequency in a discrete-time signal is $\omega = \pi$ or $f = \frac{1}{2}$, it follows that, with a sampling rate F_s, the corresponding highest values of F and Ω are

$$F_{max} = \frac{F_s}{2} = \frac{1}{2T}$$

$$\Omega_{max} = \pi F_s = \frac{\pi}{T}$$

The implications of these frequency relations can be fully appreciated by considering the two analog sinusoidal signals

$$x_1(t) = \cos 2\pi(10)t$$

$$x_2(t) = \cos 2\pi(50)t$$

which are sampled at a rate Fs = 40 Hz. The corresponding discrete-time signals or sequences are

$$x_1(n) = \cos 2\pi\left(\frac{10}{40}\right)n = \cos\frac{\pi}{2}n$$

$$x_2(n) = \cos 2\pi\left(\frac{50}{40}\right)n = \cos\frac{5\pi}{2}n$$

However, $\cos 5\pi n/2 = \cos(2\pi n + \pi n/2) = \cos \pi n/2.$

Hence $x_2(n)=x_1(n)$.

Since $x_2(t)$ yields exactly the same values as $x_1(t)$ when the two are sampled at $Fs=40$ samples per second, we say that the frequency $F_2=50$ *Hz* is an alias of the frequency $F_1 = 10$ *Hz* at the sampling rate of 40 samples per second.

F2 is not the only alias of F1.

In fact at the sampling rate of 40 samples per second, the frequency F3 = 90 Hz is also an alias of F1, as is the frequency F4 = 130 Hz, and so on.

All of the sinusoids $\cos 2\pi(F1+40k)t$, k = 1, 2, 3, 4.... sampled at 40 samples per second, yield identical values.

Consequently, they are all aliases of F1 = 10 Hz.

In general, the sampling of a continuous-time sinusoidal signal

$$x_a(t) = A\cos(2\pi F_0 t + \theta)$$

with a sampling rate $Fs = 1/T$ results in a discrete-time signal

$$x(n) = A\cos(2\pi f_0 n + \theta)$$

where $f_0 = F_o/Fs$, is the relative frequency of the sinusoid.

If we assume that $-Fs/2 < F_o < Fs/2$, the frequency f_0 of x(n) is in the range $-1/2 < f_0 < 1/2$ which is the frequency range for discrete-time signals.

In this case, the relationship between F_o and f_0 is one-to-one, and hence it is possible to identify (or reconstruct) the analog signal $x_a(t)$ from the samples x(n).

If the sinusoids

$$x_a(t) = A\cos(2\pi F_k t + \theta)$$

where

$$F_k = F_0 + kF_s, \qquad k = \pm 1, \pm 2, \ldots$$

are sampled at a rate *Fs*, it is clear that the frequency F_k is outside the fundamental frequency range —*Fs*/2 < *F* < *Fs*/2. Consequently, the sampled signal is

$$x(n) \equiv x_a(nT) = A\cos\left(2\pi \frac{F_0 + kF_s}{F_s} n + \theta\right)$$

$$= A\cos(2\pi nF_0/F_s + \theta + 2\pi kn)$$

$$= A\cos(2\pi f_0 n + \theta)$$

Thus an infinite number of continuous-time sinusoids is represented by the same discrete-time signal (i.e.. by the same set of samples).

Consequently, if we are given the sequence x(n) an ambiguity exists as to which continuous-time signal $x_a(t)$ these values represent.

The frequencies $F_k = F_0 + kFs$, $-\infty < k < \infty$ (k integer) are indistinguishable from the frequency F_0 after sampling and hence they are aliases of F_0.

The Sampling Theorem

- Given any analog signal, how should we select the sampling period T or, equivalently, the sampling rate Fs.
- we must have some information about the characteristics of the signal to be sampled.
- In particular, we must have some general information concerning the frequency content of the signal.
- For example, we know generally that the major frequency components of a speech signal fall below 3000 Hz. On the other hand, television signals, in general, contain important frequency components up to 5 MHz.
- However, if we know the maximum frequency content of the general class of signals (e.g.. the class of speech signals, the class of video signals, etc.). we can specify the sampling rate necessary to convert the analog signals to digital signals.

- Let us suppose that any analog signal can be represented as a sum of sinusoids of different amplitudes, frequencies, and phases, that is

$$x_a(t) = \sum_{i=1}^{N} A_i \cos(2\pi F_i t + \theta_i)$$

where N denotes the number of frequency components.

- Since the maximum frequency may vary slightly, we may wish to ensure that Fmax does not exceed some predetermined value by passing the analog signal through a filter that severely attenuates frequency components above Fmax.
- In practice, such filtering is commonly used prior to sampling.

- From our knowledge of F_{max}, we can select the appropriate sampling rate.
- We know that the highest frequency in an analog signal that can be unambiguously reconstructed when the signal is sampled at a rate $F_s = 1/T$, is $F_s/2$. Any frequency above $F_s/2$ or below $-F_s/2$ results in samples that are identical with a corresponding frequency in the range $-F_s/2 < F < F_s/2$.
- To avoid the ambiguities resulting from aliasing, we must select the sampling rate to be sufficiently high. That is, we must select $F_s/2$ to be greater than F_{max}. Thus to avoid the problem of aliasing, F_s is selected so that $F_s > 2F_{max}$. where Fmax is the largest frequency component in the analog signal.
- With the sampling rate selected in this manner, any frequency component, say $|F_i| < F_{max}$, in the analog signal is mapped into a discrete-time sinusoid with a frequency

$$-\frac{1}{2} \le f_i = \frac{F_i}{F_s} \le \frac{1}{2}$$

or, equivalently,

$$-\pi \le \omega_i = 2\pi f_i \le \pi$$

- Since, $|f| = 1\backslash 2$ or $|\omega| = \pi$ is the highest (unique) frequency in a discrete-time signal, the above way of choice of sampling rate avoids the problem of aliasing.
- In other words, the condition Fs > 2Fmax ensures that all the sinusoidal components in the analog signal are mapped into corresponding discrete-time frequency components with frequencies in the fundamental interval.
- Thus all the frequency components of the analog signal are represented in sampled form without ambiguity, and hence the analog signal can be reconstructed without distortion from the sample values using an "appropriate" interpolation (digital-to-analog conversion) method.
- The "appropriate" or ideal interpolation formula is specified by the sampling theorem.

- Sampling Theorem. If the highest frequency contained in an analog signal $x_a(t)$ is F_{max} = B and the signal is sampled at a rate Fs > 2 F_{max} = 2B, then $x_a(t)$ can be exactly recovered from its sample values using the interpolation function

$$g(t) = \frac{\sin 2\pi Bt}{2\pi Bt}$$

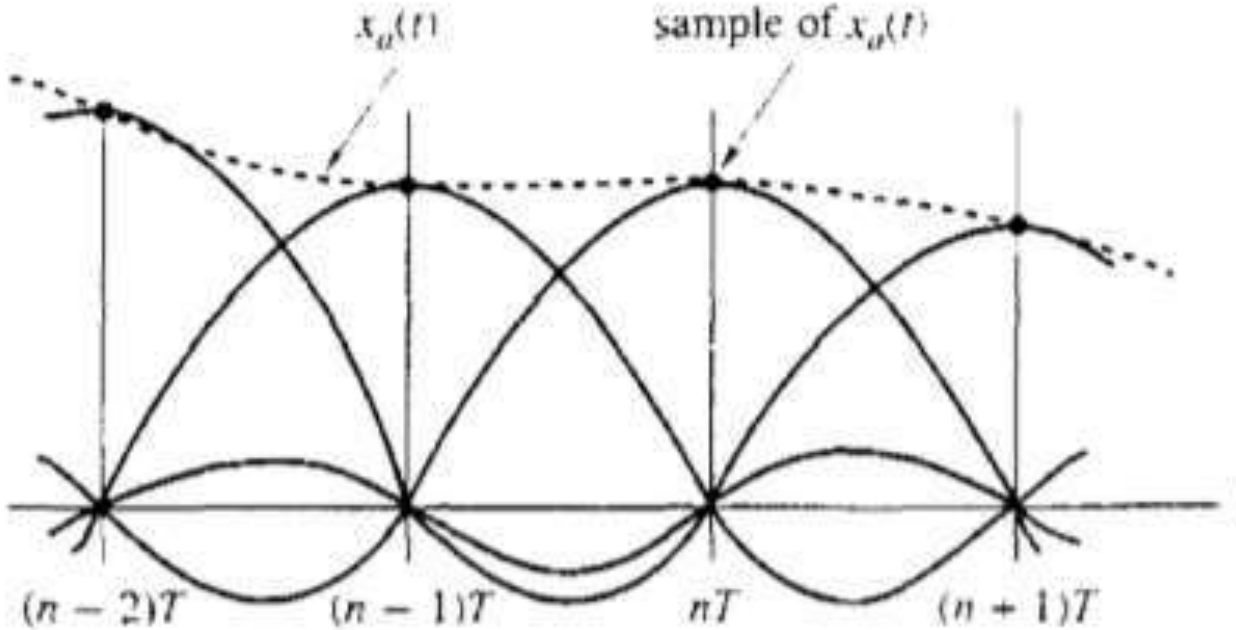
- Thus $x_a(t)$ may be expressed as

$$x_a(t) = \sum_{n=-\infty}^{\infty} x_a\left(\frac{n}{F_s}\right) g\left(t - \frac{n}{F_s}\right)$$

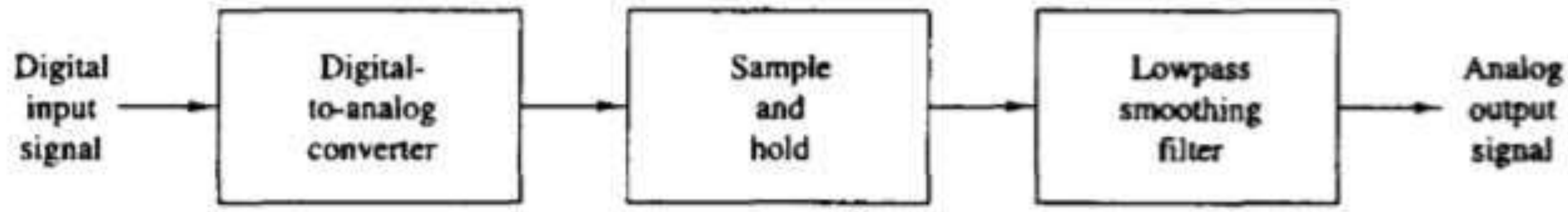
- When the sampling of $x_a(t)$ is performed at the minimum sampling rate Fs = 2B, the reconstruction formula in becomes

$$x_a(t) = \sum_{n=-\infty}^{\infty} x_a\left(\frac{n}{2B}\right) \frac{\sin 2\pi B(t - n/2B)}{2\pi B(t - n/2B)}$$

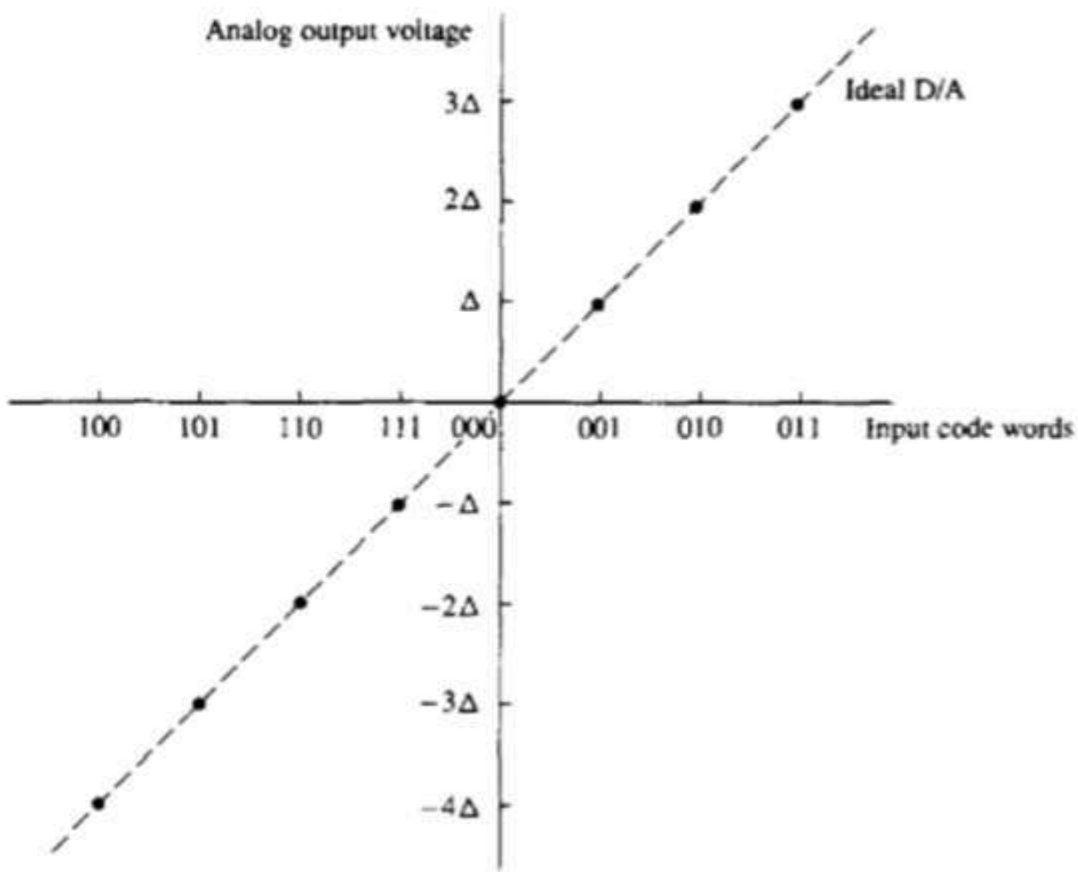
- The sampling rate Fs = 2B = 2Fmax is called the Nyquist rate.



- Figure illustrates the ideal D/A conversion process using the interpolation function *g(t)*.
- The reconstruction of $x_a(t)$ from the sequence x(n) is a complicated process, involving a weighted sum of the interpolation function g(t) and its time-shifted versions g(t—nT) for $-\infty < n < \infty$, where the weighting factors are the samples x(n).
- Because of the complexity and the infinite number of samples required, the reconstruction in practice, is usually performed by combining a D/A converter with a sample-and-hold (S/H) and a lowpass filter.

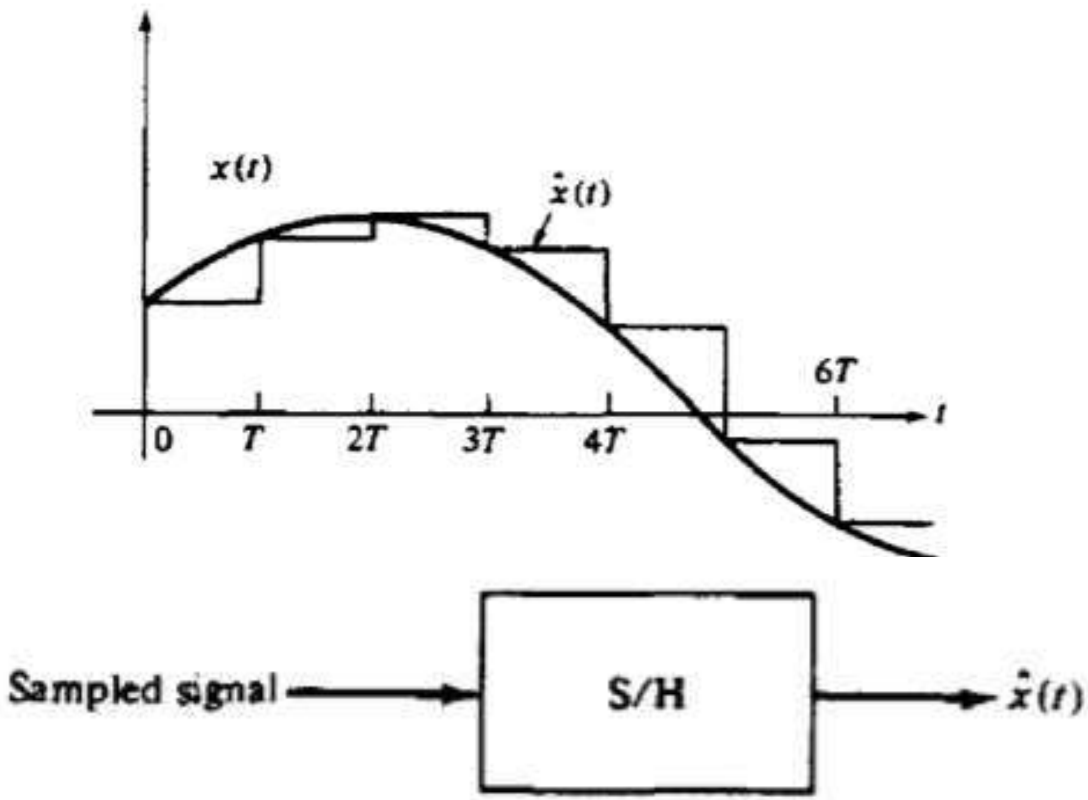


- The D/A converter accepts at its input, electrical signals that correspond to a binary word, and produces an output voltage or current that is proportional to the value of the binary word.
- Ideally, its input-output characteristic is as shown in Figure for a 3-bit bipolar signal.

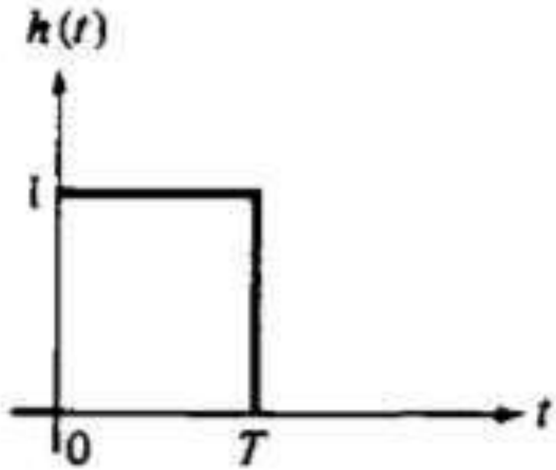


- An important parameter of a D/A converter is its settling time, which is defined as the time required for the output of the D/A converter to reach and remain within a given fraction (usually, ±1/2LSB) of the final value, after application of the input code word.
- Often, the application of the input code word results in a high-amplitude transient, called a "glitch." This is especially the case when two consecutive code words to the A/D differ by several bits.
- The usual way to remedy this problem is to use a S/H circuit designed to serve as a "deglitcher."
- Hence the basic task of the S/H is to hold the output of the D/A converter constant at the previous output value until the new sample at the output of the D/A reaches steady state, then it samples and holds the new value in the next sampling interval.
- Thus the S/H approximates the analog signal by a series of rectangular pulses whose height is equal to the corresponding value of the signal pulse.

- Figure illustrates the approximation of the analog signal x(t) by a S/H.
- As shown, the approximation, is basically a staircase function which takes the signal sample from the D/A converter and holds it for T seconds. When the next sample arrives, it jumps to the next value and holds it for T seconds, and so on.



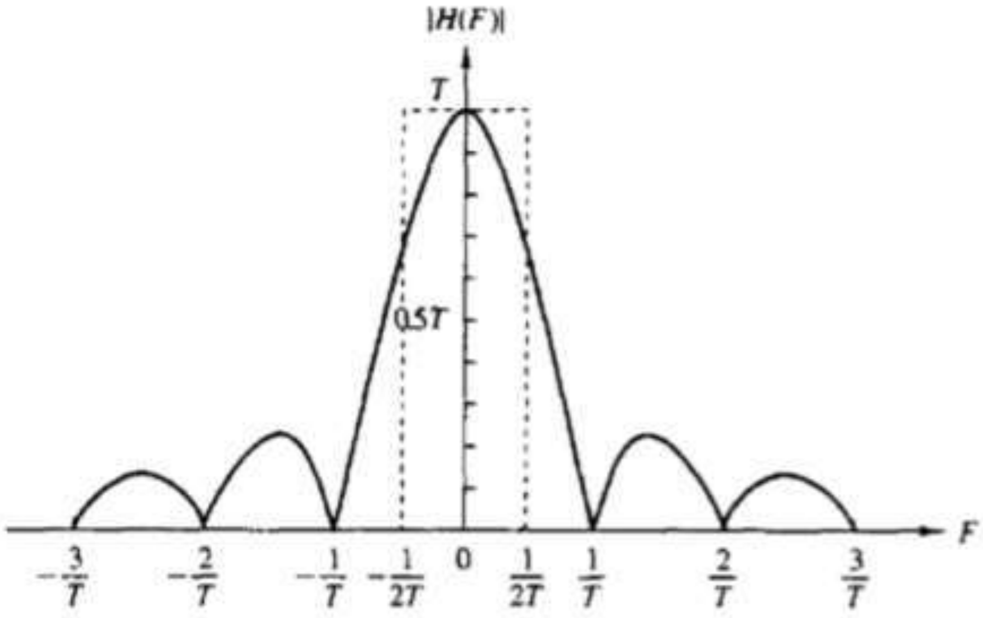
- The S/H when viewed as a linear filter, has an impulse response as shown in Figure,



$$h(t) = \begin{cases} 1, & 0 \le t \le T \\ 0, & \text{otherwise} \end{cases}$$

- The corresponding frequency response is

$$\begin{aligned} H(F) &= \int_{-\infty}^{\infty} h(t) e^{-j2\pi F t} dt \\ &= \int_{0}^{T} e^{-j2\pi F t} dt \\ &= T \left(\frac{\sin \pi F T}{\pi F T} \right) e^{-j\pi F T} \end{aligned}$$



- For comparison, the frequency response of the ideal interpolator is superimposed on the magnitude characteristics.
- It is apparent that the S/H does not possess a sharp cutoff frequency response characteristic. This is due to a large extent to the sharp transitions of its impulse response h(t).
- As a consequence, the S/H passes undesirable aliased frequency components (frequencies above Fs/2) to its output.
- To remedy this problem, it is common practice to filter these components by passing through a lowpass filter.

TABLE 4.1 FREQUENCY RANGES OF SOME BIOLOGICAL SIGNALS

Type of Signal	Frequency Range (Hz)
Electroretinogram[a]	0–20
Electronystagmogram[b]	0–20
Pneumogram[c]	0–40
Electrocardiogram (ECG)	0–100
Electroencephalogram (EEG)	0–100
Electromyogram[d]	10–200
Sphygmomanogram[e]	0–200
Speech	100–4000

[a] A graphic recording of retina characteristics.
[b] A graphic recording of involuntary movement of the eyes.
[c] A graphic recording of respiratory activity.
[d] A graphic recording of muscular action, such as muscular contraction.
[e] A recording of blood pressure.

EFFICIENT COMPUTATION OF THE DFT: FFT ALGORITHMS

- Basically, the computational problem for the DFT is to compute the sequence {X(k)} of N complex-valued numbers given another sequence of data {x(n)} of length N, according to the formula

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

where

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

- In general, the data sequence x(n) is also assumed to be complex valued. Similarly, the IDFT becomes

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

- Since the DFT and IDFT involve basically the same type of computations, our discussion of efficient computational algorithms for the DFT applies as well to the efficient computation of the IDFT.

- We observe that for each value of k, direct computation of X(k) involves N complex multiplications (4N real multiplications) and N - 1 complex additions (4N-2 real additions).
- Consequently, to compute all N values of the DFT requires N^2 complex multiplications and $N^2 - N$ complex additions.
- Direct computation of the DFT is basically inefficient primarily because it does not exploit the symmetry and periodicity properties of the phase factor W_N
- In particular, these two properties are:

Symmetry property: $W_N^{k+N/2} = -W_N^k$

Periodicity property: $W_N^{k+N} = W_N^k$

The computationally efficient algorithms, known collectively as fast Fourier transform (FFT) algorithms, exploit these two basic properties of the phase factor.

Divide-and-Conquer Approach to Computation of the DFT

- The development of computationally efficient algorithms for the DFT is made possible 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 computationally efficient algorithms known collectively as FFT algorithms.

- Let us consider the computation of an N-point DFT, where N can be factored as a product of two integers, that is,

$$N = LM$$

- The assumption that N is not a prime number is not restrictive, since we can pad any sequence with zeros to ensure a factorization of the form
- Now the sequence $x(n)$, $0 < n < N - 1$, can be stored in either a one dimensional array indexed by n or as a two-dimensional array indexed by l and m, where $0 < l < L - 1$ and $0<m<M-1$.
- Note that l is the row index and m is the column index.
- Thus, the sequence $x(n)$ can be stored in a rectangular array in a variety of ways, each of which depends on the mapping of index n to the indexes (l, m).

- When we select the mapping $n = Ml + m$.
- This leads to an arrangement in which the first row consists of the first M elements of $x(n)$, the second row consists of the next M elements of $x(n)$, and so on, as illustrated in Fig (a).
- On the other hand, the mapping $n = l + mL$ stores the first L elements of $x(n)$ in the first column, the next L elements in the second column, and so on, as illustrated in Fig (b).
- A similar arrangement can be used to store the computed DFT values.
- In particular, the mapping is from the index k to a pair of indices (p, q), where $0 < p < L - 1$ and $0 < q < M - 1$.

Row-wise $\quad n = Ml + m$

$l \backslash m$	0	1	2		$M-1$
0	$x(0)$	$x(1)$	$x(2)$	$\cdots$	$x(M-1)$
1	$x(M)$	$x(M+1)$	$x(M+2)$	$\cdots$	$x(2M-1)$
2	$x(2M)$	$x(2M+1)$	$x(2M+2)$	$\cdots$	$x(3M-1)$
	$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$
$L-1$	$x((L-1)M)$	$x((L-1)M+1)$	$x((L-1)M+2)$	$\cdots$	$x(LM-1)$

(a)

Column-wise $\quad n = l + mL$

$l \backslash m$	0	1	2		$M-1$
0	$x(0)$	$x(L)$	$x(2L)$	$\cdots$	$x((M-1)L)$
1	$x(1)$	$x(L+1)$	$x(2L+1)$	$\cdots$	$x((M-1)L+1)$
2	$x(2)$	$x(L+2)$	$x(2L+2)$	$\cdots$	$x((M-1)L+2)$
	$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$
$L-1$	$x(L-1)$	$x(2L-1)$	$x(3L-1)$	$\cdots$	$x(LM-1)$

(b)

- If we select the mapping

$$k = Mp + q$$

the DFT is stored on a row-wise basis, where the first row contains the first M elements of the DFT X(k), the second row contains the next set of M elements, and so on.

- On the other hand, the mapping k = qL + p results in a column-wise storage of X (k), where the first L elements are stored in the first column, the second set of L elements are stored in the second column, and so on.
- When x(n) is mapped into the rectangular array x(l,m) and X(k) is mapped into a corresponding rectangular array X(p, q).

Then the DFT can be expressed as a double sum over the elements of the rectangular array multiplied by the corresponding phase factors.

$$X(p,q) = \sum_{m=0}^{M-1} \sum_{l=0}^{L-1} x(l,m) W_N^{(Mp+q)(mL+l)}$$

But

$$W_N^{(Mp+q)(mL+l)} = W_N^{MLmp} W_N^{mLq} W_N^{Mpl} W_N^{lq}$$

However, $W_N^{Nmp} = 1$, $W_N^{mqL} = W_{N/L}^{mq} = W_M^{mq}$, and $W_N^{Mpl} = W_{N/M}^{pl} = W_L^{pl}$.

- With these simplifications,

$$X(p,q) = \sum_{l=0}^{L-1} \left\{ W_N^{lq} \left[\sum_{m=0}^{M-1} x(l,m) W_M^{mq} \right] \right\} W_L^{lp}$$

The expression involves the computation of DFTs of length M and length L.

- Let us subdivide the computation into three steps:

1. First, we compute the M-point DFTs for each of the rows l = 0,1........L-1, the term within square brackets.

$$F(l,q) \equiv \sum_{m=0}^{M-1} x(l,m) W_M^{mq}, \qquad 0 \le q \le M-1$$

2.Second, we compute a new rectangular array G(l,q) defined as

$$G(l,q) = W_N^{lq} F(l,q) \qquad \begin{matrix} 0 \le l \le L-1 \\ 0 \le q \le M-1 \end{matrix}$$

3. Finally, we compute the L-point DFTs for each column q = 0,1,..., M — 1, of the array G(l, q).

$$X(p,q) = \sum_{l=0}^{L-1} G(l,q) W_L^{lp}$$

The first step involves the computation of L DFTs, each of M points. Hence this step requires LM^2 complex multiplications and LM(M — 1) complex additions.

The second step requires LM complex multiplications.

Finally, the third step in the computation requires ML^2 complex multiplications and ML(L — 1) complex additions.

- Therefore, the computational complexity is

  Complex multiplications: $N(M + L + 1)$

  Complex additions: $N(M + L - 2)$

  where $N = ML$.
- Thus the number of multiplications has been reduced from N^2 to $N(M + L +1)$ and the number of additions has been reduced from $N(N - 1)$ to $N(M + L - 2)$.

When N is a highly composite number, that is, N can be factored into a product of prime numbers of the form

$$N = r1r2 \ldots r_{\gamma}$$

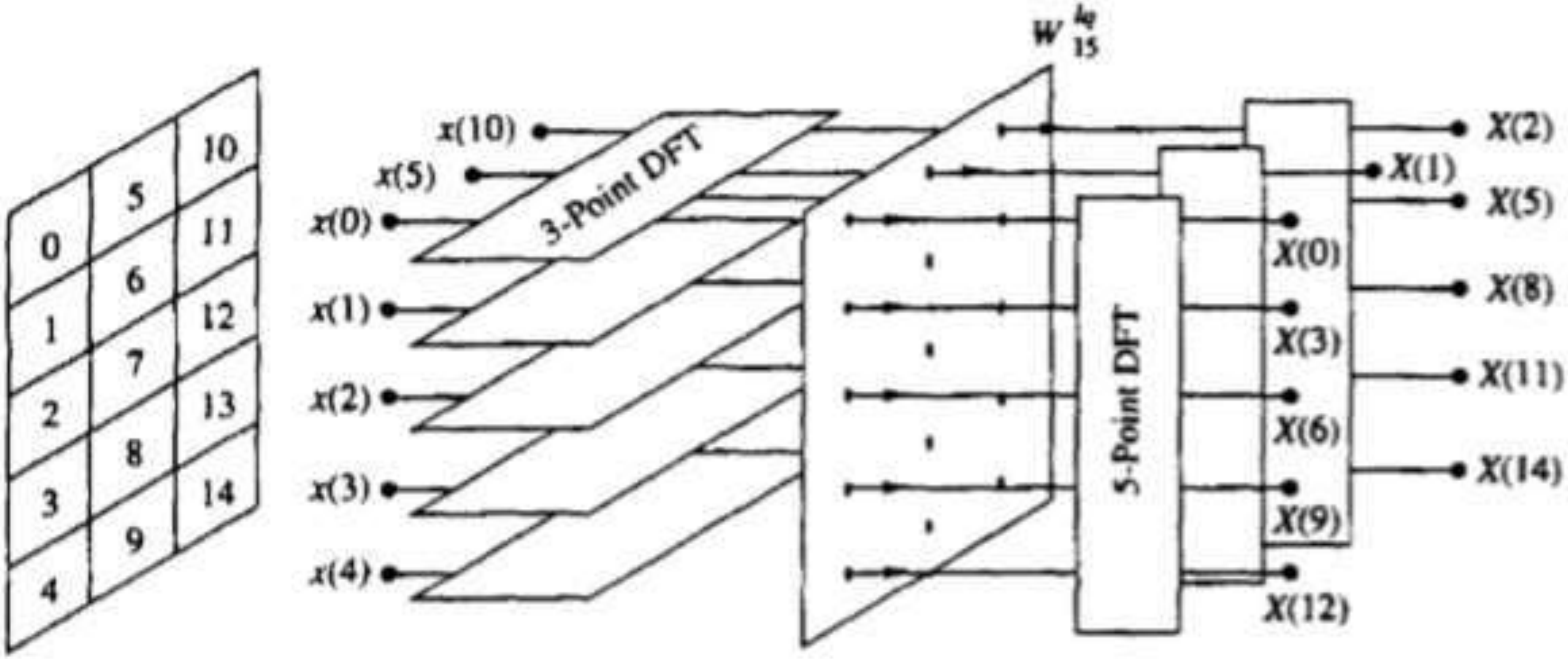
then the decomposition above can be repeated (γ -1) more times. This procedure results in smaller DFTs, which, in turn, leads to a more efficient computational algorithm.

The algorithm involves the following computations:

Algorithm 1

1. Store the signal column-wise.
2. Compute the M-point DFT of each row.
3. Multiply the resulting array by the phase factors
4. Compute the L-point DFT of each column
5. Read the resulting array row-wise.

An additional algorithm with a similar computational structure can be obtained if the input signal is stored row-wise and the resulting transformation is column-wise.



0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
x(10)
x(5)
x(0)
x(1)
x(2)
x(3)
x(4)
3-Point DFT
W_{15}^{lq}
5-Point DFT
X(2)
X(1)
X(5)
X(0)
X(8)
X(3)
X(11)
X(6)
X(14)
X(9)
X(12)

Algorithm 2

1. Store the signal row-wise.
2. Compute the L-point DFT at each column.
3. Multiply the resulting array by the phase factors.
4. Compute the M-point DFT of each row.
5. Read the resulting array column-wise.

- The two algorithms have the same complexity. However, they differ in the arrangement of the computations.
- In Radix-2 FFT Algorithms, the divide-and-conquer approach to derive fast algorithms when the size of the DFT is restricted to be a power of 2.

Radix-2 FFT Algorithms

- When N is a highly composite number, that is, N can be factored into a product of prime numbers of the form

$$N = r_1 r_2 \ldots r_\gamma$$

then the decomposition can be repeated (γ -1) times.

- This procedure results in smaller DFTs, which, in turn, leads to a more efficient computational algorithm.
- The case in which $r_1 = r_2 = \ldots r_\gamma$ so that $N = r^\gamma$
- In such a case the DFTs are of size r, so that the computation of the N-point DFT has a regular pattern.
- The number r is called the radix of the FFT algorithm.
- Let us consider the computation of the $N = 2^\gamma$ point DFT by the divide- and-conquer approach.

- When we select M = N/2 and L = 2 which results in a split of the N-point data sequence into two N/2-point data sequences f1(n) and f2(n), corresponding to the even-numbered and odd-numbered samples of x(n), respectively, that is,

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

$$f_2(n) = x(2n+1), \quad n = 0, 1, \ldots, \frac{N}{2} - 1$$

- Thus f1(n) and f2(n) are obtained by decimating x(n) by a factor of 2, and hence the resulting FFT algorithm is called a decimation-in-time algorithm.
- Now the N-point DFT can be expressed in terms of the DFTs of the decimated sequences as follows:

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

$$= \sum_{n \text{ even}} x(n) W_N^{kn} + \sum_{n \text{ odd}} x(n) W_N^{kn}$$

$$= \sum_{m=0}^{(N/2)-1} x(2m) W_N^{2mk} + \sum_{m=0}^{(N/2)-1} x(2m+1) W_N^{k(2m+1)}$$

- But $W_N^2 = W_{N/2}$, With this substitution, X(k) can be expressed as

$$X(k) = \sum_{m=0}^{(N/2)-1} f_1(m) W_{N/2}^{km} + W_N^k \sum_{m=0}^{(N/2)-1} f_2(m) W_{N/2}^{km}$$
$$= F_1(k) + W_N^k F_2(k) \qquad k = 0, 1, \ldots, N-1$$

- where F1(k) and F2(k) are the N/2-point DFTs of the sequences f1(m) and f2(m), respectively.
- Since F1(k) and F2(k) are periodic, with period N /2, we have F1(k+ N/2) = F1(k) and F2(k+ N/2) = F2(k). In addition, the factor $W_N^{k+N/2} = -W_N^k$.
- Hence

$$X(k) = F_1(k) + W_N^k F_2(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1$$
$$X\left(k + \frac{N}{2}\right) = F_1(k) - W_N^k F_2(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1$$

- The computation of X(k) requires $2(N/2)^2 + N/2 = N^2/2 + N/2$ complex multiplications. This first step results in a reduction of the number of multiplications from N^2 to $N^2/2 + N/2$, which is about a factor of 2 for N large.

we may define

$$G_1(k) = F_1(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1$$

$$G_2(k) = W_N^k F_2(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1$$

- Then the DFT X (k) may be expressed as

$$X(k) = G_1(k) + G_2(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1$$

$$X\left(k + \frac{N}{2}\right) = G_1(k) - G_2(k) \qquad k = 0, 1, \ldots, \frac{N}{2} - 1$$

- We can repeat the process for each of the sequences f1(n) and f2(n). Thus f1(n) would result in the two N/4-point sequences

$$v_{11}(n) = f_1(2n) \qquad n = 0, 1, \ldots, \frac{N}{4} - 1$$

$$v_{12}(n) = f_1(2n+1) \qquad n = 0, 1, \ldots, \frac{N}{4} - 1$$

and f2(n) would yield

$$v_{21}(n) = f_2(2n) \qquad n = 0, 1, \ldots, \frac{N}{4} - 1$$

$$v_{22}(n) = f_2(2n+1) \qquad n = 0, 1, \ldots, \frac{N}{4} - 1$$

- By computing N/4-point DFTs, we would obtain the N/2-point DFTs F1(k) and F2(k) from the relations

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

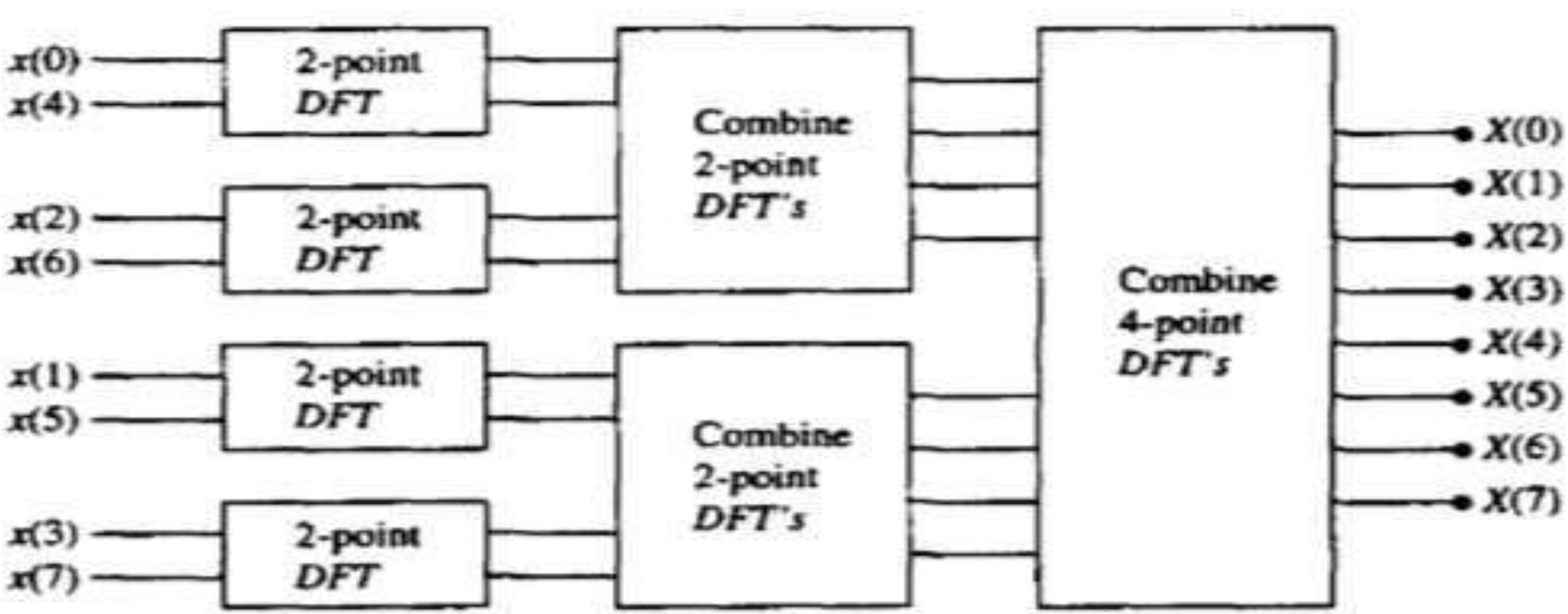
$$F_1\left(k + \frac{N}{4}\right) = V_{11}(k) - W_{N/2}^k V_{12}(k) \quad k = 0, 1, \ldots, \frac{N}{4} - 1$$

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

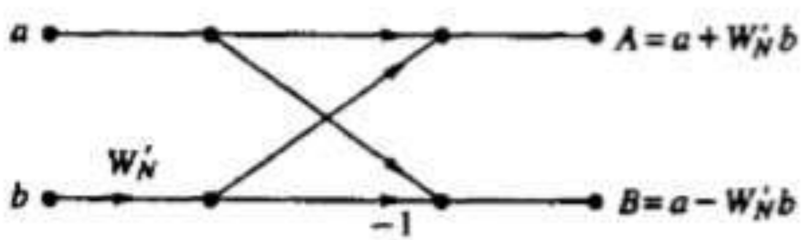
$$F_2\left(k + \frac{N}{4}\right) = V_{21}(k) - W_{N/2}^k V_{22}(k) \quad k = 0, \ldots, \frac{N}{4} - 1$$

  where the {Vij(k)} are the N/4-point DFTs of the sequences {Vij(n)}.
- The computation of {Vij(k)} requires $4(N/4)^2$ multiplications and hence the computation of F1(k) and F2(k) can be accomplished with $N^2/4 + N/2$ complex multiplications.
- An additional N/2 complex multiplications are required to compute X(k) from F1(k) and F2(k).
- Consequently, the total number of multiplications is reduced approximately by a factor of 2 again to $N^2/4 + N$.

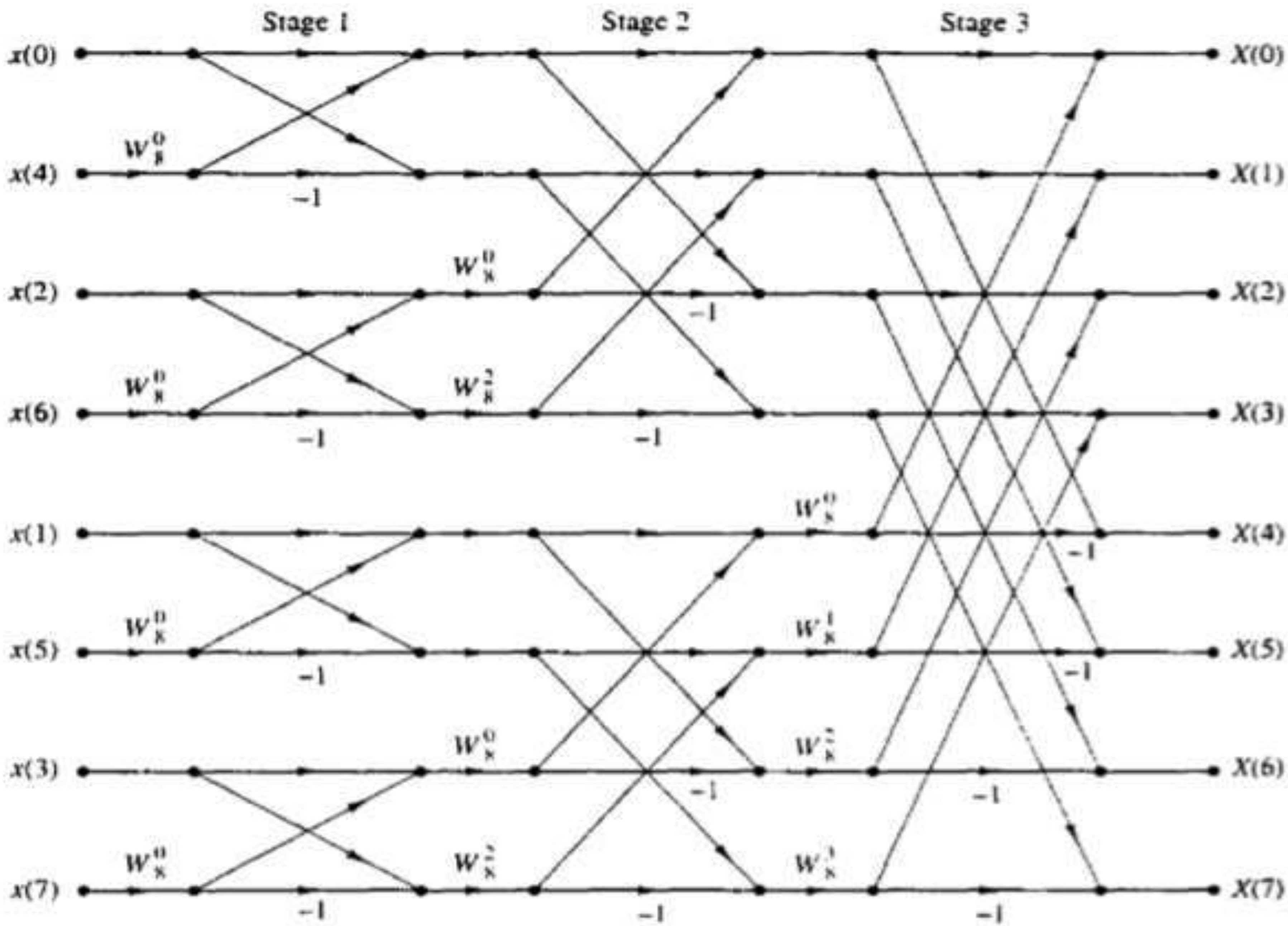
- The decimation of the data sequence can be repeated again and again until the resulting sequences are reduced to one-point sequences.
- For $N = 2^{\gamma}$, this decimation can be performed $\gamma = \log_2 N$ times.
- Figure depicts the computation of an N = 8 point DFT. The computation is performed in three stages, beginning with the computations of four two-point DFTs, then two four-point DFTs, and finally, one eight-point DFT.



- the basic computation performed at every stage, is to take two complex numbers, say the pair (a, b), multiply b by W_N^r, and then add and subtract the product from a to form two new complex numbers (A, B).
- This basic computation, which is shown in Figure, is called a butterfly because the flow graph resembles a butterfly.
- In general, each butterfly involves one complex multiplication and two complex additions.
- For $N = 2^{\gamma}$, there are N/2 butterflies per stage of the computation process and $\log_2 N$ stages. Therefore, as previously indicated the total number of complex multiplications is $(N/2)\log_2 N$ and complex additions is $N\log_2 N$



Eight point decimation-in-time FFT algorithm



Decimation-in-frequency algorithm

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

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

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

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

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

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

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

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

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

- If we define the N/2-point sequences g1(n) and g2(n) as

$$g_1(n) = x(n) + x\left(n + \frac{N}{2}\right)$$

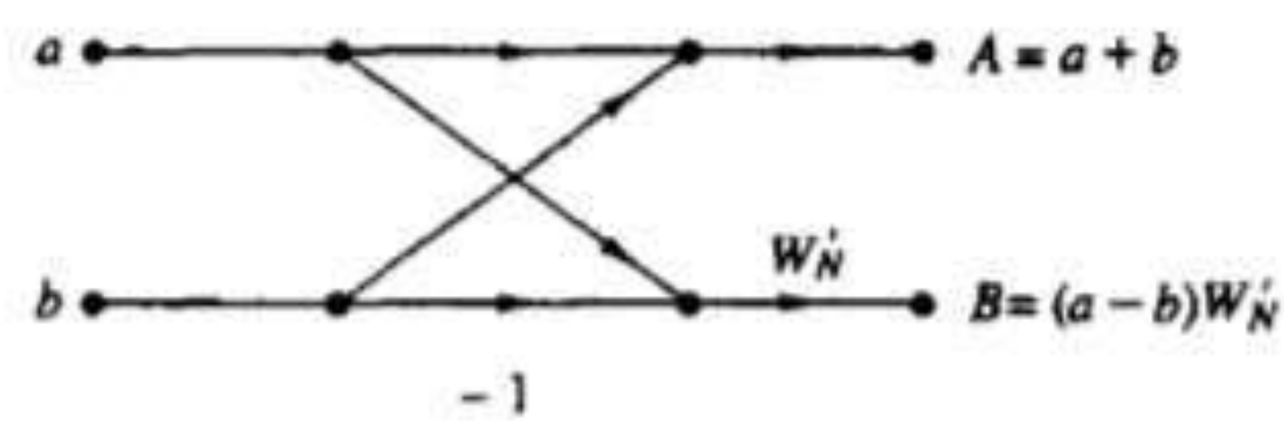
$$g_2(n) = \left[x(n) - x\left(n + \frac{N}{2}\right)\right] W_N^n \qquad n = 0, 1, 2, \ldots, \frac{N}{2} - 1$$

then

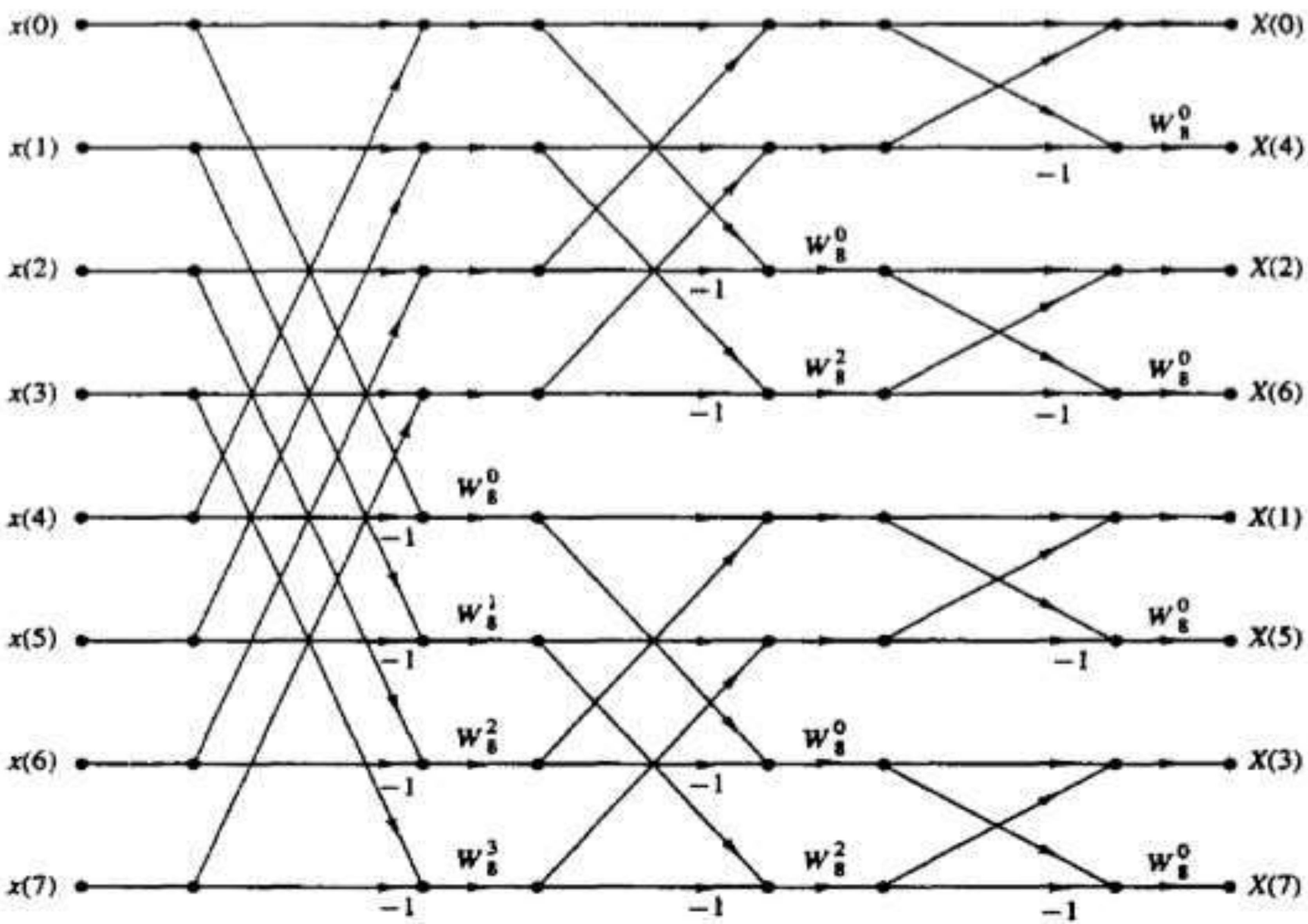
$$X(2k) = \sum_{n=0}^{(N/2)-1} g_1(n) W_{N/2}^{kn}$$

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

- This computational procedure can be repeated through decimation of the N/2-point DFTs, X(2k) and X(2k + 1). The entire process involves γ = $\log_2$ N stages of decimation, where each stage involves N/2 butterflies of the type shown in Figure.
- Consequently, the computation of the N-point DFT via the decimation- in-frequency FFT algorithm, requires (N/2) log2 N complex multiplications and $N\log_2 N$ complex additions, just as in the decimation-in-time algorithm.



Eight point decimation-in-frequency FFT algorithm



DIGITAL FILTER DESIGN

UNIT-II

IIR Filter Design by Impulse Invariance

- To design an IIR filter having a unit sample response h(n) that is the sampled version of the impulse response of the analog filter. That is,

$$h(n) \equiv h(nT) \qquad n = 0, 1, 2, \ldots$$

- when a continuous time signal xa(t) with spectrum Xa(F) is sampled at a rate Fs =1/T samples per second, the spectrum of the sampled signal is the periodic repetition of the scaled spectrum FsXa(F) with period Fs. Specifically, the relationship is

$$X(f) = F_s \sum_{k=-\infty}^{\infty} X_a[(f - k)F_s]$$

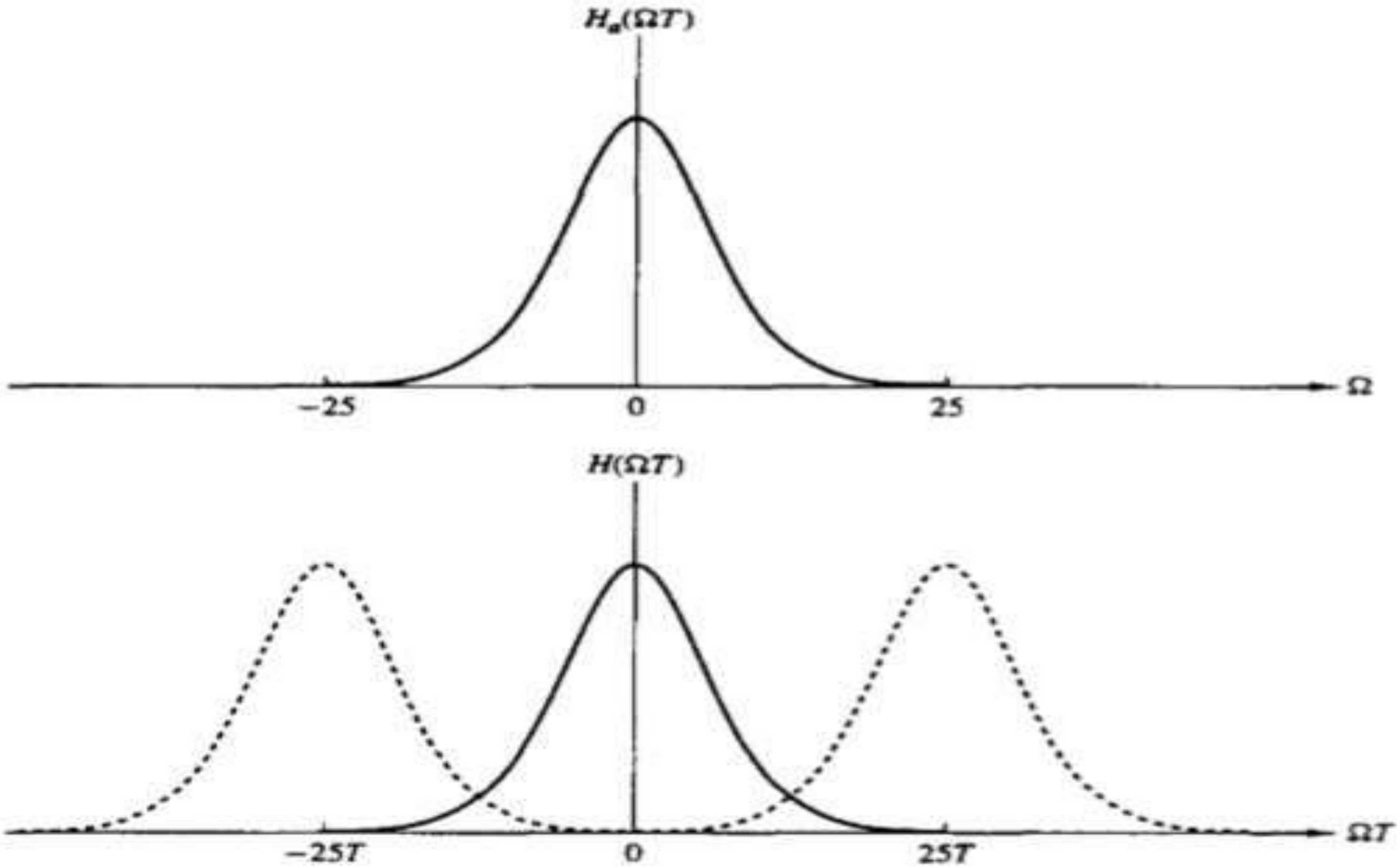
- where f = F/Fs is the normalized frequency. Aliasing occurs if the sampling rate Fs is less than twice the highest frequency contained in Xa(F).
- Sampling the impulse response of an analog filter with frequency response Ha(F), the digital filter with unit sample response h(n)=ha(nT) has the frequency response

$$H(f) = F_s \sum_{k=-\infty}^{\infty} H_a[(f - k)F_s]$$

- equivalently,

$$H(\omega) = F_s \sum_{k=-\infty}^{\infty} H_a[(\omega - 2\pi k)F_s]$$

$$H(\Omega T) = \frac{1}{T} \sum_{k=-\infty}^{\infty} H_a\left(\Omega - \frac{2\pi k}{T}\right)$$



- The digital filter with frequency response H(ω) has the frequency response characteristics of the corresponding analog filter if the sampling interval T is selected sufficiently small to completely avoid or at least minimize the effects of aliasing.
- It is also clear that the impulse invariance method is inappropriate for designing highpass filters due the to spectrum aliasing that results from the sampling process.
- To investigate the mapping of points between the z-plane and the s-plane implied by the sampling process,

$$H(z)|_{z=e^{sT}} = \frac{1}{T} \sum_{k=-\infty}^{\infty} H_a\left(s - j\frac{2\pi k}{T}\right)$$

where

$$H(z) = \sum_{n=0}^{\infty} h(n) z^{-n}$$

$$H(z)|_{z=e^{sT}} = \sum_{n=0}^{\infty} h(n) e^{-sTn}$$

- Let us consider the mapping of points from the s-plane to the z-plane implied by the relation

$$z = e^{sT}$$

- If we substitute s = $\sigma + j\Omega$ and express the complex variable z in polar form as z = $re^{j\omega}$, the above equation becomes

$$re^{j\omega} = e^{\sigma T} e^{j\Omega T}$$

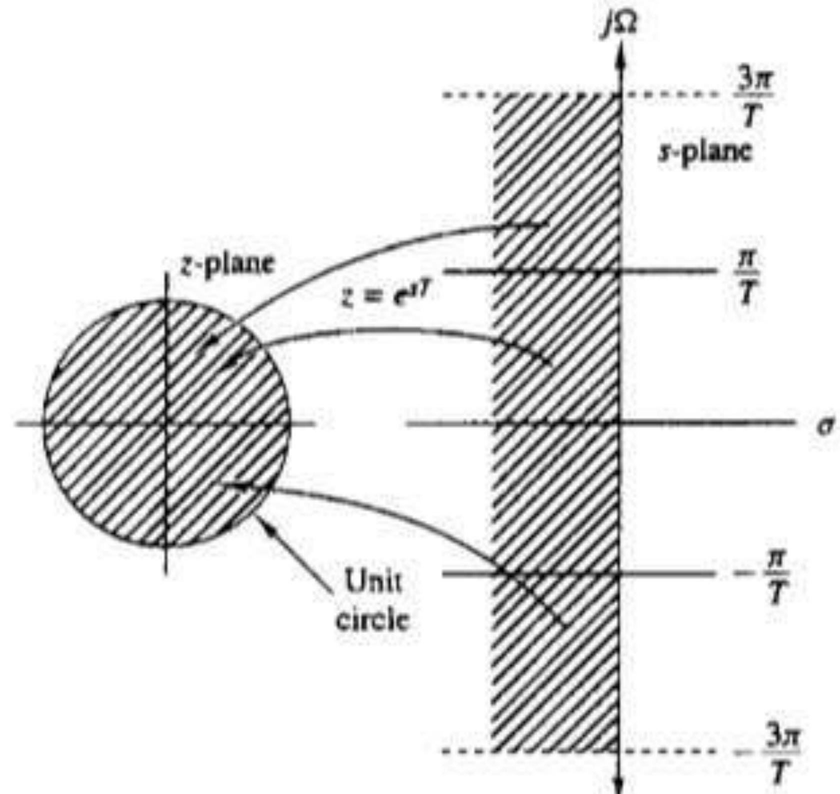
Clearly we have

$$r = e^{\sigma T}$$

$$\omega = \Omega T$$

- Consequently, $\sigma < 0$ implies that $0 < r < 1$ and $\sigma > 0$ implies that $r > 1$. When $\sigma = 0$, we have $r = 1$. Therefore, the LHP in s is mapped inside the unit circle in z and the RHP in s is mapped outside the unit circle in z.
- Also, the $j\Omega$ -axis is mapped into the unit circle in z as indicated.
- However, the mapping of the $j\Omega$ -axis into the unit circle is not one-to-one.

- Since ω is unique over the range $(-\pi, \pi)$, the mapping $\omega = \Omega T$ implies that the interval $-\pi/T < \Omega < \pi/T$ maps into the corresponding values of $-\pi < \omega < \pi$.
- Furthermore, the frequency interval $\pi/T < \Omega < 3\pi/T$ also maps into the interval $-\pi < \omega < \pi$ and, in general, so does the interval $(2k - 1)\pi/T < \Omega < (2k+1)\pi/T$, when k is an integer.
- Thus the mapping from the analog frequency Ω to the frequency variable ω in the digital domain is many-to-one, which simply reflects the effects of aliasing due to sampling.



- The system function of the analog filter in partial-fraction form on the assumption that the poles of the analog filter are distinct

$$H_a(s) = \sum_{k=1}^{N} \frac{c_k}{s - p_k}$$

where $\{p_k\}$ are the poles of the analog filter and $\{c_k\}$ are the coefficients in the partial-fraction expansion.

$$h_a(t) = \sum_{k=1}^{N} c_k e^{p_k t} \qquad t \geq 0$$

If we sample ha(t) periodically at t = nT, we have $h(n) = h_a(nT)$

$$= \sum_{k=1}^{N} c_k e^{p_k T n}$$

Now, with the substitution of h(n), the system function of the resulting digital IIR filter becomes

$$H(z) = \sum_{n=0}^{\infty} h(n) z^{-n}$$

$$= \sum_{n=0}^{\infty} \left(\sum_{k=1}^{N} c_k e^{p_k T n} \right) z^{-n}$$

$$= \sum_{k=1}^{N} c_k \sum_{n=0}^{\infty} (e^{p_k T} z^{-1})^n$$

- The inner sum converges because $p_k < 0$ and yields

$$\sum_{n=0}^{\infty}(e^{p_k T} z^{-1})^n = \frac{1}{1 - e^{p_k T} z^{-1}}$$

- Therefore, the system function of the digital filter is

$$H(z) = \sum_{k=1}^{N} \frac{c_k}{1 - e^{p_k T} z^{-1}}$$

We observe that the digital filter has poles at

$$z_k = e^{p_k T} \qquad k = 1, 2, \dots, N$$

- Although the poles are mapped from the s-plane to the z-plane by the above relationship, we should emphasize that the zeros in the two domains do not satisfy the same relationship. The development that resulted in H(z) was based on a filter having distinct poles. It can be generalized to include multiple-order poles.

EXAMPLE 8.4 For the analog transfer function

$$H_a(s) = \frac{2}{(s+1)(s+3)}$$

determine $H(z)$ if (a) $T = 1$ s and (b) $T = 0.5$ s using impulse invariant method.

Solution: Given, $H_a(s) = \dfrac{2}{(s+1)(s+3)}$

Using partial fractions, $H_a(s)$ can be expressed as:

$$H_a(s) = \frac{A}{s+1} + \frac{B}{s+3}$$

$$A = (s+1)\, H_a(s)\big|_{s=-1} = \left.\frac{2}{s+3}\right|_{s=-1} = 1$$

$$B = (s+3)\, H_a(s)\big|_{s=-3} = \left.\frac{2}{s+1}\right|_{s=-3} = -1$$

$\therefore$

$$H_a(s) = \frac{1}{s+1} - \frac{1}{s+3} = \frac{1}{s-(-1)} - \frac{1}{s-(-3)}$$

By impulse invariant transformation, we know that

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

Here $H_a(s)$ has two poles and $p_1 = -1$ and $p_2 = -3$.

Therefore, the system function of the digital filter is:

$$H(z) = \frac{1}{1-e^{p_1 T} z^{-1}} - \frac{1}{1-e^{p_2 T} z^{-1}}$$

$$= \frac{1}{1-e^{-T} z^{-1}} - \frac{1}{1-e^{-3T} z^{-1}}$$

(a) When $T = 1$ s

$$H(z) = \frac{1}{1 - e^{-1}z^{-1}} - \frac{1}{1 - e^{-3}z^{-1}}$$

$$= \frac{1}{1 - 0.3678z^{-1}} - \frac{1}{1 - 0.0497z^{-1}}$$

$$= \frac{(1 - 0.0497z^{-1}) - (1 - 0.3678z^{-1})}{(1 - 0.3678z^{-1})(1 - 0.0497z^{-1})}$$

$$= \frac{0.3181z^{-1}}{1 - 0.4175z^{-1} + 0.0182z^{-2}}$$

(b) When $T = 0.5$ s

$$H(z) = \frac{1}{1 - e^{-0.5}z^{-1}} - \frac{1}{1 - e^{-3\times 0.5}z^{-1}}$$

$$= \frac{1}{1 - 0.606z^{-1}} - \frac{1}{1 - 0.223z^{-1}}$$

$$= \frac{(1 - 0.223z^{-1}) - (1 - 0.606z^{-1})}{(1 - 0.606z^{-1})(1 - 0.223z^{-1})}$$

$$= \frac{0.383z^{-1}}{1 - 0.829z^{-1} + 0.135z^{-2}}$$

EXAMPLE 8.7 The system function of an analog filter is expressed as:

$$H_a(s) = \frac{2}{s(s+2)}$$

Find the corresponding $H(z)$ using the impulse invariant method for a sampling frequency of 4 samples per second.

Solution: Given sampling rate = 4 samples/second

$\therefore$ Sampling period $T = \frac{1}{4} = 0.25$ s

Expressing the given $H_a(s)$ in terms of partial fractions, we have

$$H_a(s) = \frac{2}{s(s+2)} = \frac{1}{s} - \frac{1}{s+2} = \frac{1}{s-(0)} - \frac{1}{s-(-2)}$$

By the impulse invariant transformation, we know that

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

Here $H_a(s)$ has two poles and $p_1 = 0$ and $p_2 = -2$.

Therefore, the system function of the digital filter is:

$$H(z) = \frac{1}{1-e^{p_1T}z^{-1}} - \frac{1}{1-e^{p_2T}z^{-1}}$$

$$= \frac{1}{1-e^{(0)T}z^{-1}} - \frac{1}{1-e^{(-2)T}z^{-1}}$$

$$= \frac{1}{1-z^{-1}} - \frac{1}{1-e^{-2(0.25)}z^{-1}}$$

$$= \frac{1}{1-z^{-1}} - \frac{1}{1-0.606\, z^{-1}}$$

$$= \frac{(1-0.606z^{-1})-(1-z^{-1})}{(1-z^{-1})(1-0.606z^{-1})}$$

$$= \frac{0.394z^{-1}}{1-1.606z^{-1}+0.606z^{-2}}$$

Bilinear transformation method

- Impulse invariant transformation method. is appropriate only for the design of low-pass filters and band pass filters whose resonant frequencies are small. These techniques are not suitable for high-pass or band reject filters.
- The limitation is overcome in the mapping technique called the bilinear transformation.
- This transformation is a one-to-one mapping from the s-domain to the z-domain.
- That is, 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.
- So the transformation of a stable analog filter results in a stable digital filter.

Let the system function of the analog filter be $H_a(s) = \dfrac{b}{s+a}$

The differential equation describing the above analog filter can be obtained as:

$$H_a(s) = \frac{Y(s)}{X(s)} = \frac{b}{s+a}$$

or

$$sY(s) + aY(s) = bX(s)$$

Taking inverse Laplace transform on both sides, we get

$$\frac{dy(t)}{dt} + a\,y(t) = bx(t)$$

Integrating the above equation between the limits $(nT - T)$ and nT, we have

$$\int_{nT-T}^{nT} \frac{dy(t)}{dt}\,dt + a\int_{nT-T}^{nT} y(t)\,dt = b\int_{nT-T}^{T} x(t)\,dt$$

The trapezoidal rule for numeric integration is expressed as:

$$\int_{nT-T}^{nT} a(t)\,dt = \frac{T}{2}[a(nT) + a(nT-T)]$$

Therefore, we get

$$y(nT) - y(nT-T) + a\frac{T}{2}y(nT) + a\frac{T}{2}y(nT-T) = b\frac{T}{2}x(nT) + b\frac{T}{2}x(nT-T)$$

Taking z-transform, we get

$$Y(z)[1-z^{-1}]+a\frac{T}{2}[1+z^{-1}]Y(z)=b\frac{T}{2}[1+z^{-1}]X(z)$$

Therefore, the system function of the digital filter is:

$$\frac{Y(z)}{X(z)}=H(z)=\frac{b}{\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}+a}$$

Comparing this with the analog filter system function $H_a(s)$ we get

$$s=\frac{2}{T}\left(\frac{1-z^{-1}}{1+z^{-1}}\right)=\frac{2}{T}\left(\frac{z-1}{z+1}\right)$$

Rearranging, we can get

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

This is the relation between analog and digital poles in bilinear transformation. So to convert an analog filter function into an equivalent digital filter function, just put

$$s=\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}\text{ in }H_a(s)$$

The general characteristic of the mapping $z = e^{sT}$ may be obtained by putting $s = \sigma + j\Omega$ and expressing the complex variable z in the polar form as $z = re^{j\omega}$ in the above equation for s.

Thus,
$$s = \frac{2}{T}\left(\frac{z-1}{z+1}\right) = \frac{2}{T}\left(\frac{re^{j\omega}-1}{re^{j\omega}+1}\right)$$

or
$$s = \frac{2}{T}\frac{(re^{j\omega}-1)(re^{-j\omega}+1)}{(re^{j\omega}+1)(re^{-j\omega}+1)} = \frac{2}{T}\left[\frac{r^2-1}{1+r^2+2r\cos\omega} + j\frac{2r\sin\omega}{1+r^2+2r\cos\omega}\right]$$

Since $s = \sigma + j\Omega$, we get

$$\sigma = \frac{2}{T}\left[\frac{r^2-1}{1+r^2+2r\cos\omega}\right]$$

and
$$\Omega = \frac{2}{T}\left[\frac{2r\sin\omega}{1+r^2+2r\cos\omega}\right]$$

From the above equation for σ, we observe that if $r < 1$ then $\sigma < 0$ and if $r > 1$, then $\sigma > 0$, and if $r = 1$, then $\sigma = 0$. Hence the left half of the s-plane maps into points inside the unit circle in the z-plane, the right half of the s-plane maps into points outside the unit circle in the z-plane and the imaginary axis of s-plane maps into the unit circle in the z-plane. This transformation results in a stable digital system.

Relation between analog and digital frequencies

On the imaginary axis of s-plane $\sigma = 0$ and correspondingly in the z-plane $r = 1$.

$$\therefore \quad \Omega = \frac{2}{T}\left(\frac{2\sin\omega}{1+1+2\cos\omega}\right) = \frac{2}{T}\left(\frac{\sin\omega}{1+\cos\omega}\right)$$

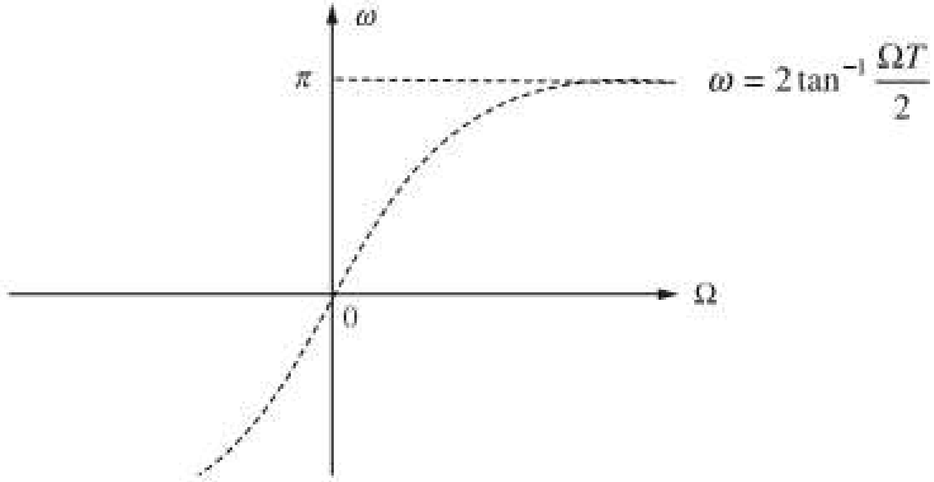
$$= \frac{2}{T}\left(\frac{2\sin\frac{\omega}{2}\cos\frac{\omega}{2}}{1+2\cos^2\omega/2-1}\right) = \frac{2}{T}\tan\frac{\omega}{2}$$

$\therefore$ The relation between analog and digital frequencies is:

$$\Omega = \frac{2}{T}\tan\frac{\omega}{2}$$

or equivalently, we have $\omega = 2\tan^{-1}\frac{\Omega T}{2}$.

The above relation between analog and digital frequencies shows that the entire range in Ω is mapped only once into the range $-\pi \le \omega \le \pi$. The entire negative imaginary axis in the s-plane (from $\Omega = -\infty$ to 0) is mapped into the lower half of the unit circle in z-plane (from $\omega = -\pi$ to 0) and the entire positive imaginary axis in the s-plane (from $\Omega = \infty$ to 0) is mapped into the upper half of unit circle in z-plane (from $\omega = 0$ to $+\pi$).



- the mapping is non-linear and the lower frequencies in analog domain are expanded in the digital domain, whereas the higher frequencies are compressed.
- This is due to the nonlinearity of the arctangent function and usually known as frequency warping.
- In designing digital filter using bilinear transformation, the effect of warping on amplitude response can be eliminated by prewarping the analog filter.
- In this method, the specified digital frequencies are converted to analog equivalent and these analog frequencies are called prewarp frequencies.
- Using the prewarp frequencies, the analog filter transfer function is designed, and then it is transformed to digital filter transfer function.

EXAMPLE 8.10 Convert the following analog filter with transfer function

$$H_a(s) = \frac{s + 0.1}{(s + 0.1)^2 + 9}$$

into a digital IIR filter by using bilinear transformation. The digital IIR filter is having a resonant frequency of $\omega_r = \pi/2$.

Solution: From the transfer function, we observe that $\Omega_c = 3$. The sampling period T can be determined using the equation:

$$\Omega_c = \frac{2}{T} \tan \frac{\omega_r}{2}$$

$$\therefore \qquad T = \frac{2}{\Omega_c} \tan \frac{\omega_r}{2} = \frac{2}{3} \tan \frac{\pi/2}{2} = 0.6666 \text{ s}$$

Using the bilinear transformation, the digital filter system function is:

$$H(z) = H_a(s)\Big|_{s=\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}} = H_a(s)\Big|_{s=3\frac{1-z^{-1}}{1+z^{-1}}}$$

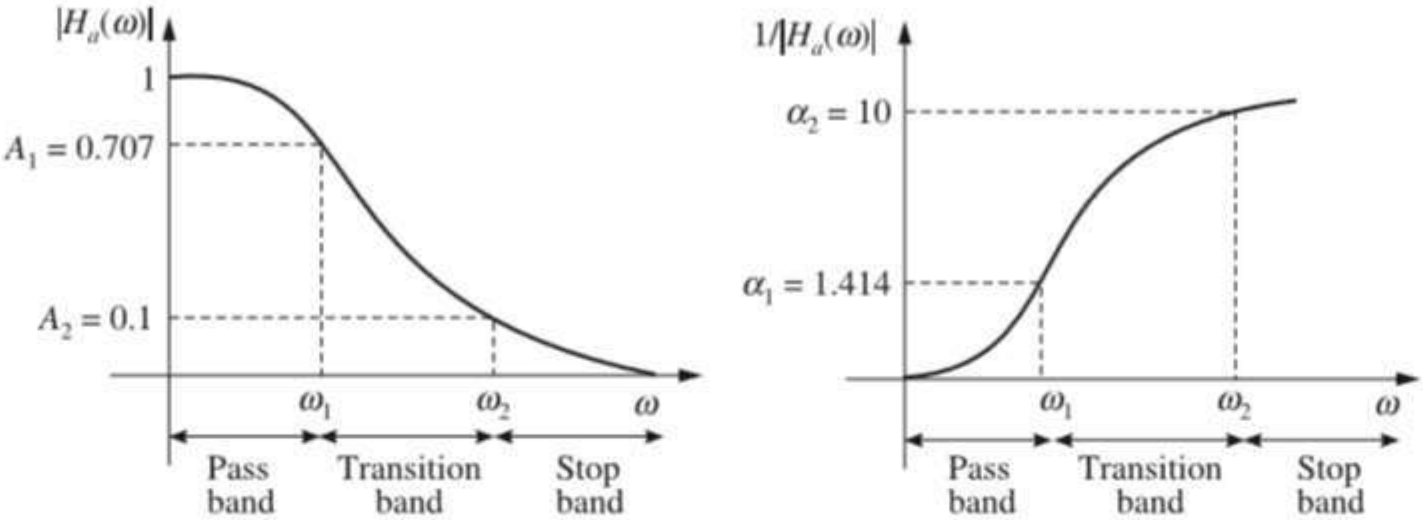
$$\therefore \qquad H(z) = \frac{s + 0.1}{(s + 0.1)^2 + 9}\Bigg|_{s=3\frac{1-z^{-1}}{1+z^{-1}}}$$

$$= \frac{3\dfrac{1-z^{-1}}{1+z^{-1}} + 0.1}{\left[3\dfrac{1-z^{-1}}{1+z^{-1}} + 0.1\right]^2 + 9}$$

$$= \frac{\left[3(1-z^{-1}) + 0.1(1+z^{-1})\right][1+z^{-1}]}{\left[3(1-z^{-1}) + 0.1(1+z^{-1})\right]^2 + 9(1+z^{-1})^2}$$

$$= \frac{3.1 + 0.2z^{-1} - 2.9z^{-2}}{18.61 + 0.02z^{-1} + 17.41z^{-2}}$$

THE MAGNITUDE RESPONSE OF LOW-PASS FILTER IN TERMS OF GAIN AND ATTENUATION



Let ω_1 = Passband frequency in rad/s.
ω_2 = Stopband frequency in rad/s.

Let the gain at the passband frequency ω_1 be A_1 and the gain at the stopband frequency ω_2 be A_2, i.e.

$$A_1 = \left|H(\omega)\right|_{\omega=\omega_1} \quad \text{and} \quad A_2 = \left|H(\omega)\right|_{\omega=\omega_2}$$

The filter may be expressed in terms of the gain or attenuation at the edge frequencies. Let α_1 be the attenuation at the passband edge frequency ω_1, and α_2 be the attenuation at the stopband edge frequency ω_2.

i.e.
$$\alpha_1 = \frac{1}{A_1} = \frac{1}{\left|H(\omega)\right|_{\omega=\omega_1}} \quad \text{and} \quad \alpha_2 = \frac{1}{A_2} = \frac{1}{\left|H(\omega)\right|_{\omega=\omega_2}}$$

The maximum value of normalized gain is unity, so A_1 and A_2 are less than 1 and α_1 and α_2 are greater than 1.

Another popular unit that is used for filter specification is dB. When the gain is expressed in dB, it will be a negative dB. When the attenuation is expressed in dB, it will be a positive dB.

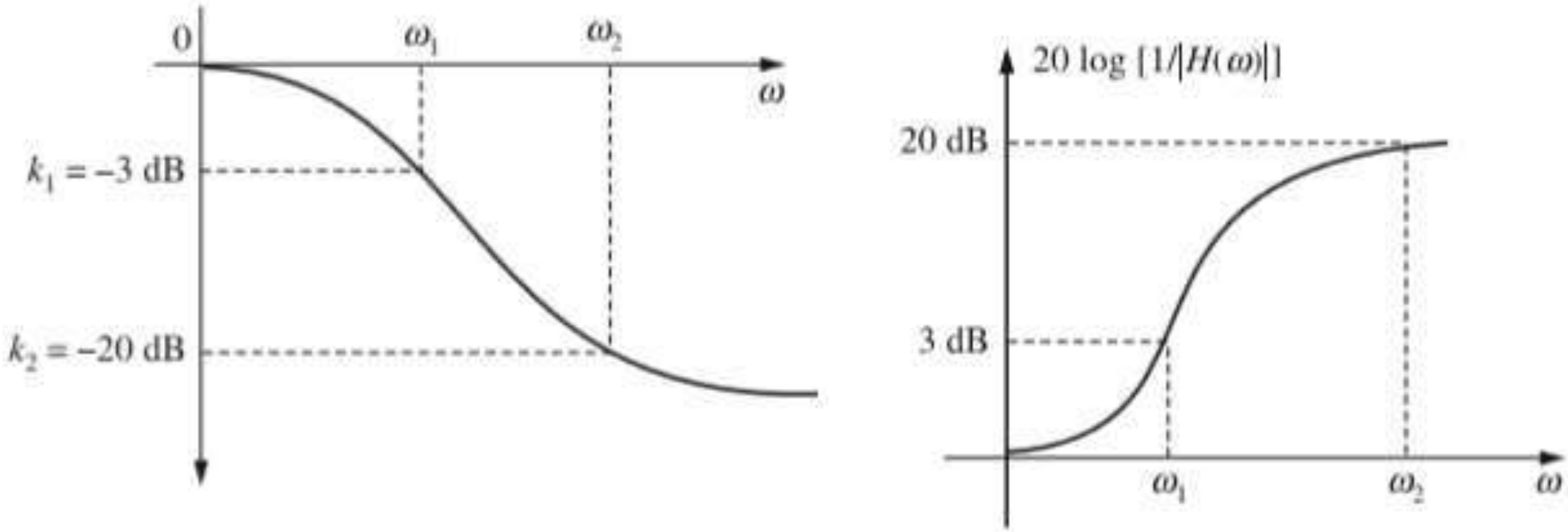
Let k_1 = Gain in dB at a passband frequency ω_1

k_2 = Gain in dB at a stopband frequency ω_2

The gain can be converted into normal values as follows:

$$\begin{aligned} 20 \log A_1 &= k_1 \\ \log A_1 &= k_1/20 \\ A_1 &= 10^{k_1/20} \end{aligned} \qquad \Bigg| \qquad \begin{aligned} 20 \log A_2 &= k_2 \\ \log A_2 &= k_2/20 \\ A_2 &= 10^{k_2/20} \end{aligned}$$

When expressed in dB, the gain and attenuation will have only change in sign because $\log \alpha = \log(1/A) = -\log A$. (Hence when dB is positive it is attenuation and when dB is negative it is gain).



Sometimes the specifications are given in terms of passband ripple δ_p and stopband ripple δ_s. In this case, the dB gain and attenuation can be estimated as follows:

$$k_1 = 20 \log (1 - \delta_p) \quad \alpha_1 = -20 \log (1 - \delta_p)$$

$$k_2 = 20 \log \delta_s \quad \alpha_2 = -20 \log \delta_s$$

If the ripples are specified in dB, then the minimum passband ripple is equal to k_1 and the negative of maximum passband attenuation is equal to k_2.

DESIGN OF LOW-PASS DIGITAL BUTTERWORTH FILTER

- To design a Butterworth IIR digital filter, first an analog Butterworth filter transfer function is determined using the given specifications.
- Then the analog filter transfer function is converted to a digital filter transfer function using either impulse invariant transformation or bilinear transformation.
- The analog Butterworth filter is designed by approximating the ideal frequency response using an error function. The error function is selected such that the magnitude is maximally flat in the passband and monotonically decreasing in the stopband.
- The magnitude response of low-pass filter obtained by this approximation is given by

where Ωc is the 3 dB cutoff frequency and N is the order of the filter.

Design procedure for low-pass digital Butterworth IIR filter

The low-pass digital Butterworth filter is designed as per the following steps:

Let A_1 = Gain at a passband frequency ω_1
A_2 = Gain at a stopband frequency ω_2
Ω_1 = Analog frequency corresponding to ω_1
Ω_2 = Analog frequency corresponding to ω_2

Step 1 Choose the type of transformation, i.e., either bilinear or impulse invariant transformation.

Step 2 Calculate the ratio of analog edge frequencies Ω_2/Ω_1.

For bilinear transformation

$$\Omega_1 = \frac{2}{T}\tan\frac{\omega_1}{2}, \Omega_2 = \frac{2}{T}\tan\frac{\omega_2}{2} \quad \therefore \frac{\Omega_2}{\Omega_1} = \frac{\tan \omega_2/2}{\tan \omega_1/2}$$

For impulse invariant transformation,

$$\Omega_1 = \frac{\omega_1}{T}, \quad \Omega_2 = \frac{\omega_2}{T} \quad \therefore \frac{\Omega_2}{\Omega_1} = \frac{\omega_2}{\omega_1}$$

Step 3 Decide the order N of the filter. The order N should be such that

$$N \geq \frac{1}{2}\frac{\log\left\{\left[\frac{1}{A_2^2}-1\right]\Big/\left[\frac{1}{A_1^2}-1\right]\right\}}{\log\frac{\Omega_2}{\Omega_1}}$$

Choose N such that it is an integer just greater than or equal to the value obtained above.

Step 4 Calculate the analog cutoff frequency $\Omega_c = \dfrac{\Omega_1}{\left[\dfrac{1}{A_1^2} - 1\right]^{1/2N}}$

For bilinear transformation $\Omega_c = \dfrac{\dfrac{2}{T}\tan \omega_1/2}{\left[\dfrac{1}{A_1^2} - 1\right]^{1/2N}}$

For impulse invariant transformation $\Omega_c = \dfrac{\omega_1/T}{\left[\dfrac{1}{A_1^2} - 1\right]^{1/2N}}$

Step 5 Determine the transfer function of the analog filter.
Let $H_a(s)$ be the transfer function of the analog filter. When the order N is even, for unity dc gain filter, $H_a(s)$ is given by

$$H_a(s) = \prod_{k=1}^{N/2} \frac{\Omega_c^2}{s^2 + b_k\Omega_c s + \Omega_c^2}$$

When the order N is odd, for unity dc gain filter, $H_a(s)$ is given by

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

The coefficient b_k is given by

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

For normalized case, $\Omega_c = 1$ rad/s

Step 6 Using the chosen transformation, transform the analog filter transfer function $H_a(s)$ to digital filter transfer function $H(z)$.

Step 7 Realize the digital filter transfer function $H(z)$ by a suitable structure.

Poles of the normalized Butterworth filter

The Butterworth low-pass filter has a magnitude squared response given by

$$|H_a(\omega)|^2 = \frac{1}{1 + \left(\frac{\Omega}{\Omega_c}\right)^{2N}}$$