ECE 844 Unit 11

Chapter 6 – Structure for Discrete-Time Systems

Review of Direct Form I and Direct Form II

Start with the most basic form of a linear difference equation associated with a digital filter:

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

A block diagram of a system that could be used to implement this difference equation is shown below. This structure is called Direct Form I.

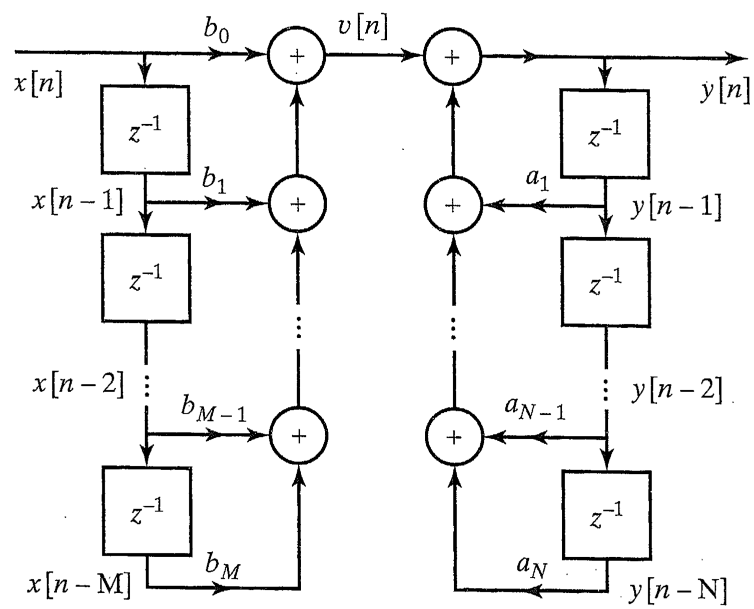


Figure 6.3 Block diagram representation for a general Nth-difference equation.

The corresponding system function is

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

Note that we could write H(z) as

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

where

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

and

$$H_2(z) = \frac{1}{1 - \sum_{k=1}^{N} a_k z^{-k}}.$$

Note that in the previous figure (showing Direct Form I), the input is first passed through the zeros of the original H(z), the then through the poles. If we denote the output of the first subsystem (representing the zeros) as v(n), then the original difference equation can represented by the combination of the following two difference equations:

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

Note that H(z) can be represented equivalently as

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

That means that the overall system function is the same if we pass the input first through the poles of the system, then through the zeros.

A block diagram for this form of implementation is shown below:

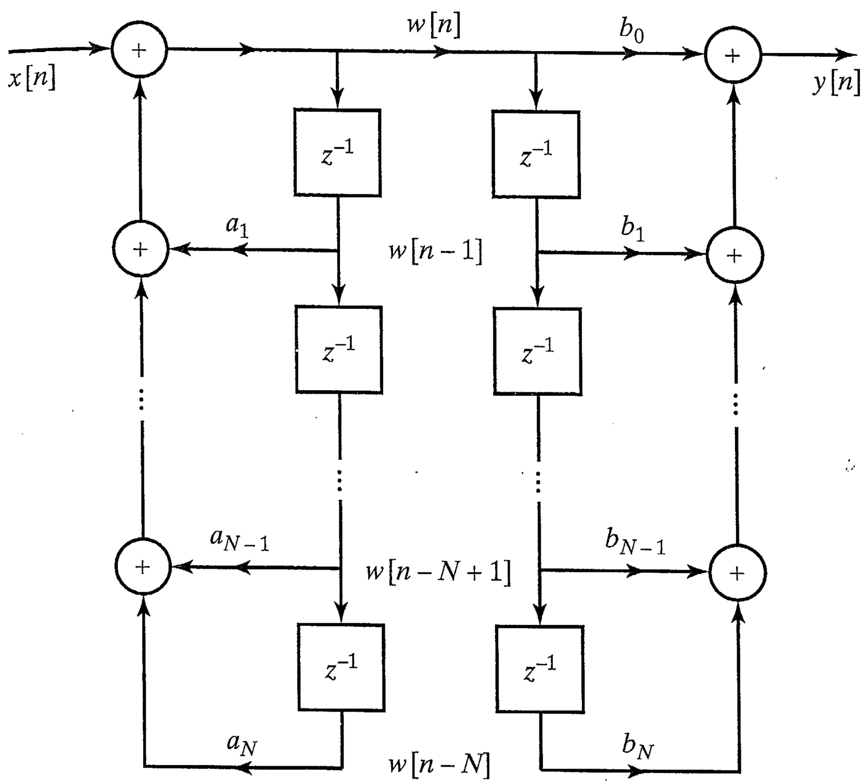


Figure 6.4 Rearrangement of block diagram of Figure 6.3. We assume for convenience that $N = M$. If $N \neq M$, some of the coefficients will be zero.

Note that the two columns of delay elements perform the same operations on the same input signal. Therefore, the implementation can be simplified by combining the two delay elements in the same row, resulting in the configuration shown below:

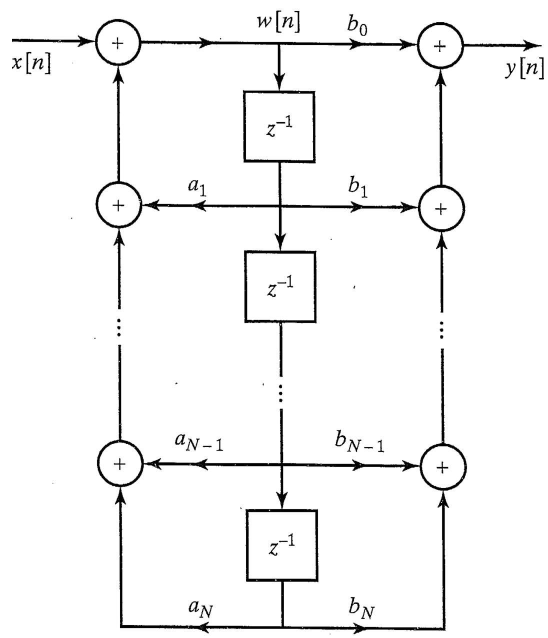


Figure 6.5 Combination of delays in Figure 6.4.

The above implementation is called Direct Form II. If the output of the first subsystem (the poles) is called w(n), the overall system can now be represented by the following pair of difference equations:

$$w(n) = x(n) + \sum_{k=1}^{N} a_k w(n-k) \quad \text{and} \quad y(n) = \sum_{k=0}^{M} b_k w(n-k).$$

Signal Flow Graph Representation of Systems

- Similar to Block Diagrams
- Formally defined as a network of directed branches that connect at nodes.
- In a linear signal flow graph, the output of each branch is a linear transformation of the branch input, such as multiplication by a constant or a simple delay
- The output of each node is the sum of the outputs of all branches entering that node.

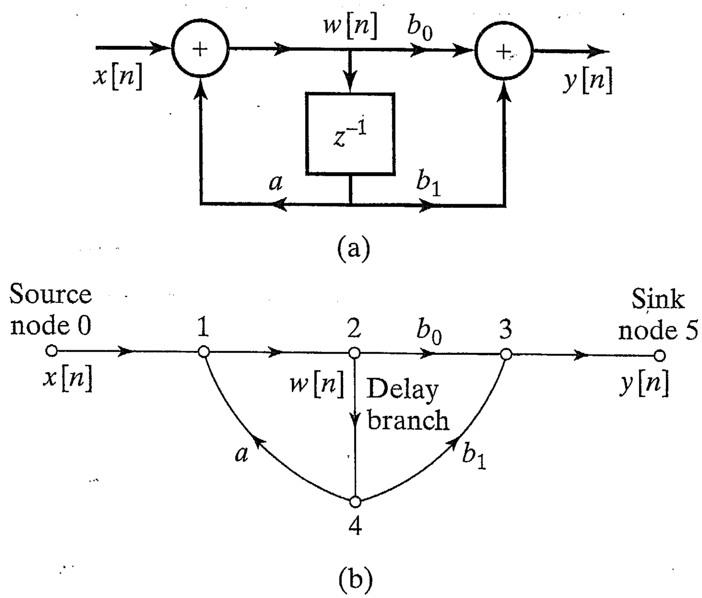


Figure 6.10 (a) Block diagram representation of a first-order digital filter. (b) Structure of the signal flow graph corresponding to the block diagram in (a).

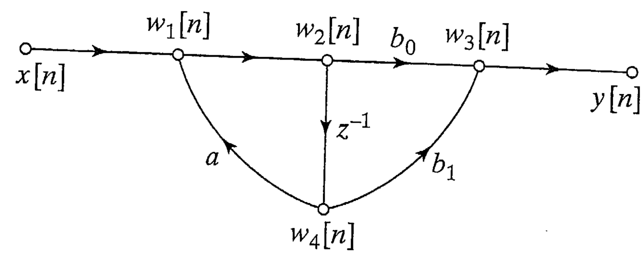


Figure 6.11 Signal flow graph of Figure 6.10(b) with the delay branch indicated by z^{-1}.

The computation needed to generate each new value of the output y(n) is shown below:

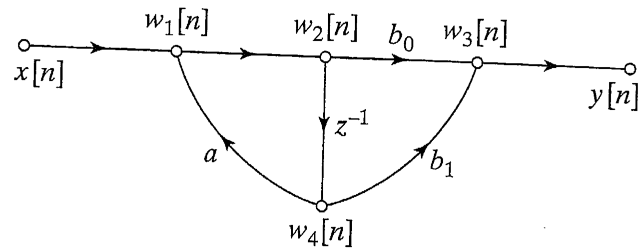
Step 1: $w_1(n) = aw_4(n) + x(n)$

Step 2: $w_2(n) = w_1(n)$

Step 3: $w_3(n) = b_0 w_2(n) + b_1 w_4(n)$

Step 4: $y(n) = w_3(n)$

Step 5: $w_4(n) = w_2(n-1)$



Note that <u>the order of the above computation steps is important</u>. The above steps should be implemented in the order shown, with the following exceptions:

- Steps 4 and 5 could be interchanged.
- Step 5 could be always be evaluated first.

<u>General Rules for Evaluating Node Values in Block Diagrams or Signal Flow Diagrams</u>

- First update all nodes which are not the output of a delay branch, using the following rule:

 -Update a node only if all other nodes (except for nodes that are outputs of delay units) which feed this node have already been updated.

- Then update nodes which are at the output of delay branches, using the following rule:

 -Update a node at the output of a delay branch only if all other nodes <u>which it feeds</u> through delays have already been updated.

Note: The above rules assume that there is at least one delay branch in each loop.

Examples of Possible Order of Computation

One possible order of computation:

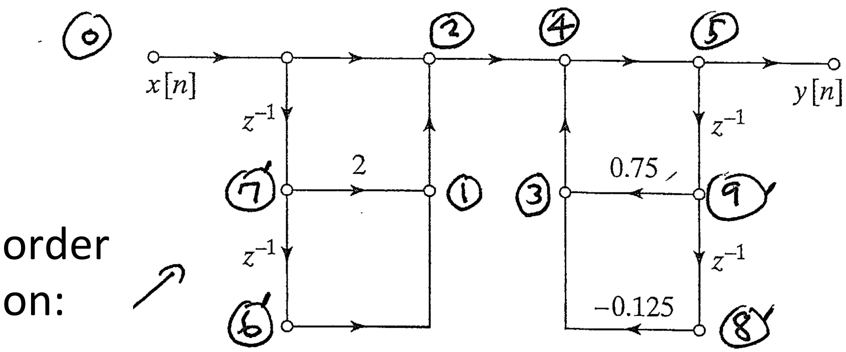


Figure 6.16 Direct form I structure for Example 6.4.

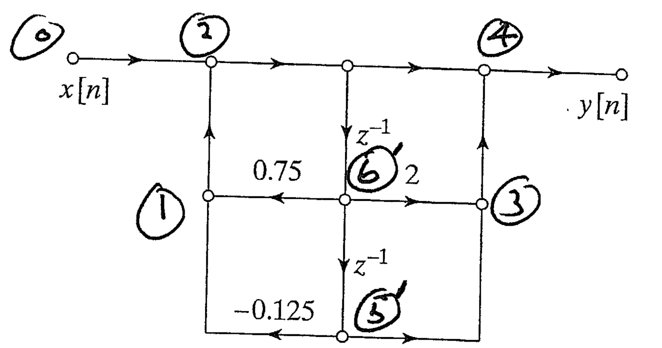


Figure 6.17 Direct form II structure for Example 6.4.

Another Example: Order of Updating Nodes

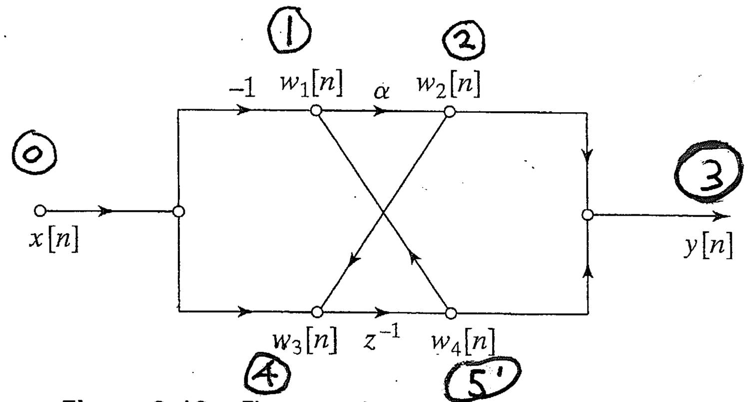


Figure 6.12 Flow graph not in standard direct form.

Determination of the System Function From the Flow Graph of Figure 6.12

Node equations:

$w_1(n) = w_4(n) - x(n)$

$w_2(n) = \alpha\, w_1(n)$

$w_3(n) = w_2(n) + x(n)$

$w_4(n) = w_3(n-1)$

$y(n) = w_2(n) + w_4(n)$

Corresponding z-domain equations:

$W_1(z) = W_4(z) - X(z)$

$W_2(z) = \alpha W_1(z)$

$W_3(z) = W_2(z) + X(z)$

$W_4(z) = z^{-1} W_3(z)$

$Y(z) = W_2(z) + W_4(z)$

To obtain an expression for H(z), we can combine the above equations so as to eliminate $W_1(z), W_2(z), W_3(z),$ and $W_4(z)$, leaving only $X(z)$ and $Y(z)$ and system parameters (e.g., α and z^{-1}).

First, eliminate $W_1(z)$ by substituting the equation for $W_1(z)$ into the equation for $W_2(z)$:

$$W_2(z) = \alpha W_1(z) = \alpha\left[W_4(z) - X(z)\right].$$

Next, eliminate $W_3(z)$ by substituting the equation for $W_3(z)$ into the equation for $W_4(z)$:

$$W_4(z) = z^{-1}W_3(z) = z^{-1}\left[W_2(z) + X(z)\right].$$

Now substitute the new expression for $W_4(z)$ into the new expression for $W_2(z)$:

$$W_2(z) = \alpha\left[W_4(z) - X(z)\right] = \alpha\left\{z^{-1}\left[W_2(z) + X(z)\right] - X(z)\right\}$$

$$= \alpha z^{-1}W_2(z) + \alpha z^{-1}X(z) - \alpha X(z).$$

$$W_2(z)\left[1 - \alpha z^{-1}\right] = \alpha X(z)\left[z^{-1} - 1\right]$$

$$W_2(z) = \frac{\alpha\left[z^{-1} - 1\right]}{\left[1 - \alpha z^{-1}\right]}X(z). \qquad \text{(equation 6.23a)}$$

Now use this expression for $W_2(z)$ to solve for $W_4(z)$:

$$W_4(z) = z^{-1}\left[W_2(z) + X(z)\right] = z^{-1}\left\{\frac{\alpha\left[z^{-1} - 1\right]}{\left[1 - \alpha z^{-1}\right]}X(z) + X(z)\right\}$$

$$= z^{-1}\left[\frac{\alpha z^{-1} - \alpha + 1 - \alpha z^{-1}}{1 - \alpha z^{-1}}\right]X(z)$$

$$= z^{-1}\frac{(-\alpha + 1)}{1 - \alpha z^{-1}}X(z). \qquad \text{(equation 6.23b)}$$

Finally, we substitute the expressions for $W_2(z)$ and $W_4(z)$ into the expression for $Y(z)$:

$$\begin{aligned} Y(z) &= W_2(z) + W_4(z) \\ &= \frac{\alpha\left[z^{-1}-1\right]}{\left[1-\alpha z^{-1}\right]} X(z) + z^{-1}\frac{(-\alpha+1)}{1-\alpha z^{-1}} X(z) \\ &= \left(\frac{z^{-1}-\alpha}{1-\alpha z^{-1}}\right) X(z). \end{aligned}$$

The signal flow graph for a Direct Form I implementation of this reduced version of H(z) is shown below:

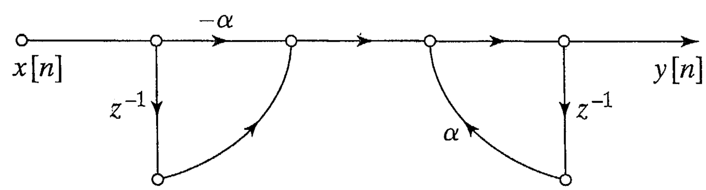


Figure 6.13 Direct form I equivalent of Figure 6.12.

Section 6.3 Basic Structures for IIR Systems

Already covered:

Direct Form I (pass signal though zeros first, then through poles)

Direct Form II (pass signal though poles first, then through zeros)

Due to considerations of quantization error and scaling, high order IIR filters are normally implemented using a cascade or parallel implementation of second and first order sections.

Cascade Form for Implementing H(z)

In order to obtain a cascade implementation of an IIR filter, H(z) is first expressed as

$$H(z) = A\frac{\prod_{k=1}^{M_1}(1-f_k z^{-1})\prod_{k=1}^{M_2}(1-g_k z^{-1})(1-g_k^* z^{-1})}{\prod_{k=1}^{N_1}(1-c_k z^{-1})\prod_{k=1}^{N_2}(1-d_k z^{-1})(1-d_k^* z^{-1})}.$$

A block diagram showing a cascade implementation of a 6-th order IIR system is shown below. This figure includes 3 second-order sections, each implemented using Direct Form II.

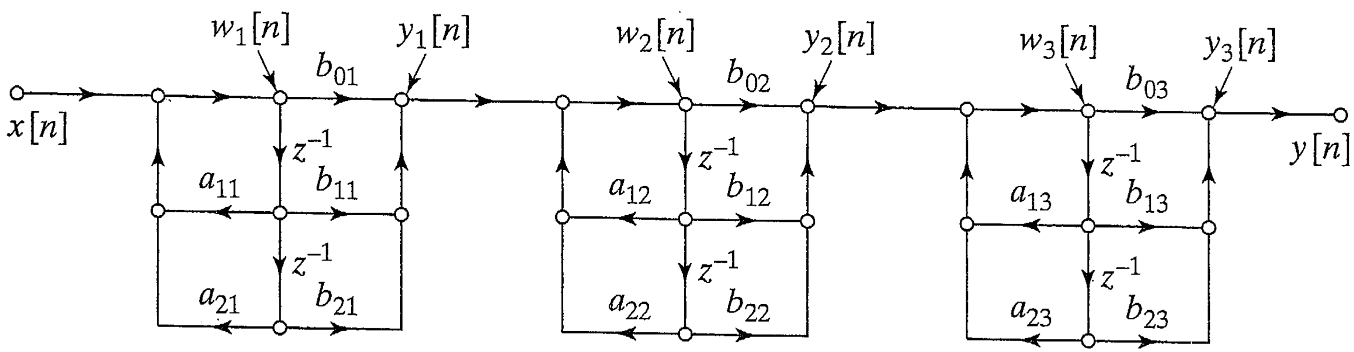
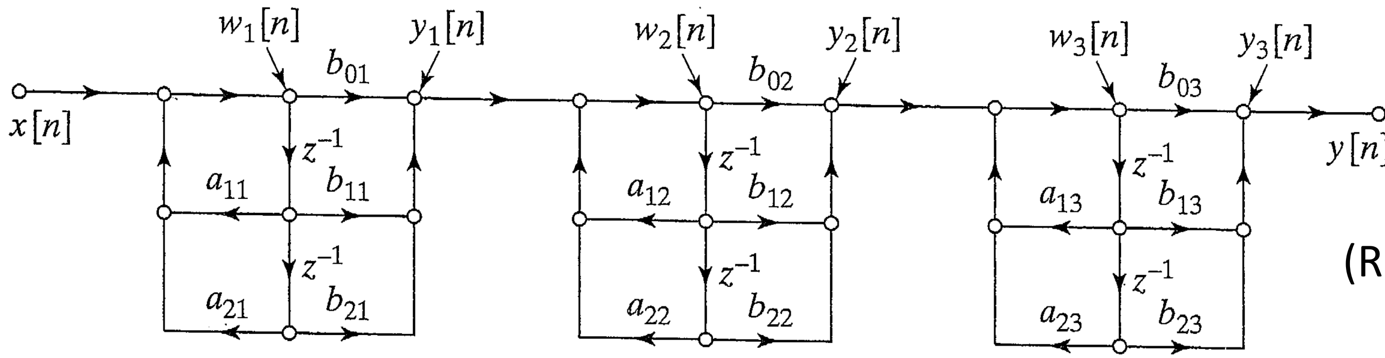


Figure 6.18 Cascade structure for a 6th order system with a direct form II Realization of each 2nd order section.



(Repeated figure)

Figure 6.18 Cascade structure for a 6th order system with a direct form II Realization of each 2nd order section.

The set of difference equations that could be used to implement a digital filter represented as a cascade of N_s second order sections is shown below:

$$y_0(n) = x(n)$$

$$w_k(n) = a_{1k} w_k(n-1) + a_{2k} w_k(n-2) + y_{k-1}(n) \qquad k = 1, 2, \ldots, N_s$$

$$y_k(n) = b_{0k} w_k(n) + b_{1k} w_k(n-1) + b_{1k} w_k(n-2) \qquad k = 1, 2, \ldots, N_s$$

$$y(n) = y_{N_s}(n).$$

• If H(z) consists of a cascade of N_s second order sections, the number of possible ways to pair N_s pole sections with N_s zero sections is $(N_s)!$.

• For each pairing of pole sections with zero sections, there are $(N_s)!$ ways to order these pairs, from the input toward the output of the filter.

Therefore, the total number different combinations of pairings and orderings is $[(N_s)!]^2$.

<u>Note</u>: If we didn't limit ourselves to using Direct Form II for each second order section, even more implementations are possible.

If all real poles and zeros, as well as complex poles and zeros, are combined into second order factors, the cascade form can be represented as

$$H(z) = \prod_{k=1}^{N_s} \frac{(b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2})}{(1 - a_{1k} z^{-1} - a_{2k} z^{-2})}.$$

Note that implementing H(z) in the form would require <u>5 multiplications per section</u>.

More generally, if the number of zeros (M) equals the number of poles (N) and N is an even number , then $N_s = N/2$ and number of multiplications required to generate each new output is $5\,N_s = 5(N/2) = (2.5)N$. In comparison, implementation of Direct Form I or Direct Form II in unfactored form requires $2N + 1$ multiplications per each new output.

Another way to represent second-order factors for the cascade form is shown below:

$$H(z) = b_0 \prod_{k=1}^{N_s} \frac{(1 + \tilde{b}_{1k} z^{-1} + \tilde{b}_{2k} z^{-2})}{(1 - a_{1k} z^{-1} - a_{2k} z^{-2})}.$$

If this version of H(z) is used, the total number of multiplications needed per output is 4 Ns +1 = 2N + 1. (Note that the scale factors for the second order sections are combined to create a single, overall scale factor.)

This assumes that the scale factors for all 2nd order sections are combined to create a single overall scale factor.) However, the “5-multiplications per section” version is often preferred because it permits distributing the gain of the system, which is often helpful to dealing with scaling issues, as will be discussed shortly.

Parallel Forms

Second order sections can also be combined in a parallel configuration, where the second order sections are based on using a partial fraction representation of H(z), as shown below:

$$H(z) = \sum_{k=0}^{N_p} C_k z^{-k} + \sum_{k=1}^{N_1} \frac{A_k}{1 - c_k z^{-1}} + \sum_{k=1}^{N_2} \frac{B_k(1 - e_k z^{-1})}{(1 - d_k z^{-1})(1 - d_k^* z^{-1})}$$

where $N_p = M - N$ and where M is the total number of zeros and N is the total number of poles. (If $N_p < 0$, the first summation is not included in the expression above.)

A parallel implementation of a 6-th order IIR filter is shown below:

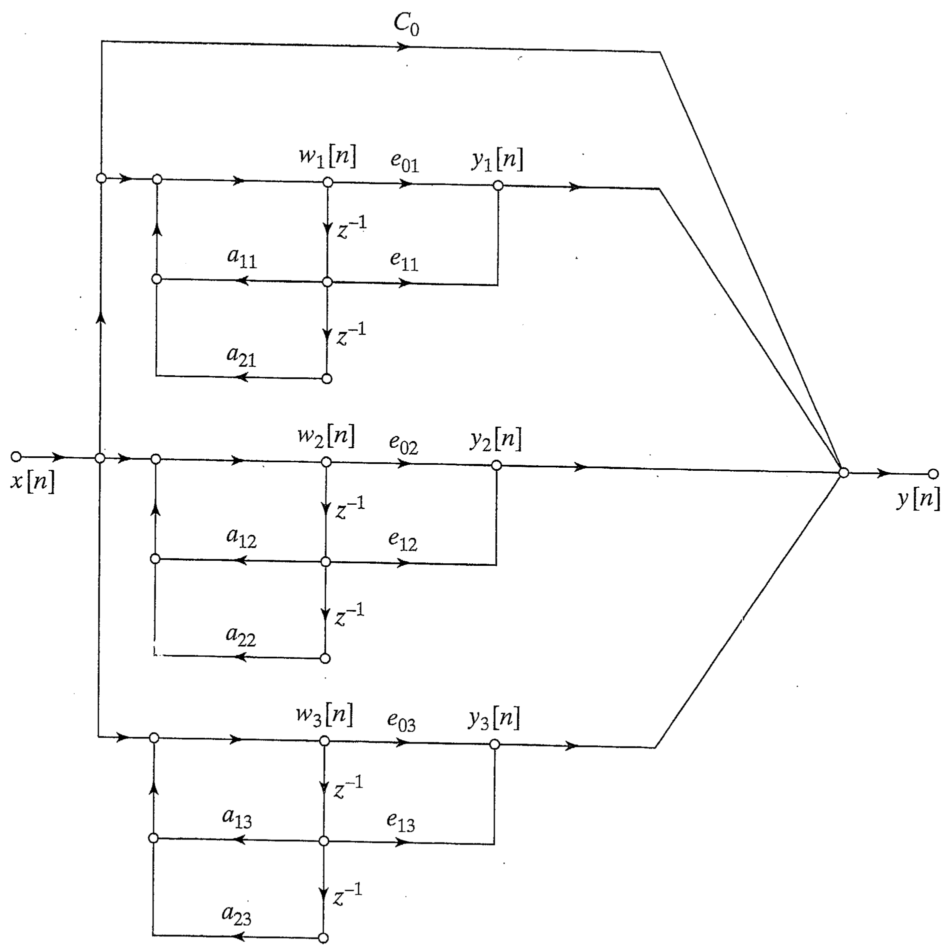


Figure 6.20 Parallel form structure for a 6th order system (M = N = 6) with the real and complex poles grouped in pairs.

A general representation of the difference equations that could be used to implement a parallel configuration of second-order Direct Form II sections is shown below:

$$w_k(n) = a_{1k}w_k(n-1) + a_{2k}w_k(n-2) + x(n), \quad k=1,2,\ldots,N_s$$

$$y_k(n) = e_{0k}w_k(n) + e_{1k}w_k(n-1), \quad k=1,2,\ldots,N_s$$

$$y(n) = \sum_{k=0}^{N_p} C_k x(n-k) + \sum_{k=1}^{N_s} y_k(n). \quad \text{(equation 6.36c)}$$

(If M < N, the summation involving C_k is not included.)

<u>Transposed Forms</u>

A new implementation of a system having the same overall H(z) can be obtained by creating the transposed form of the system. The step involved as follows:

1. Reverse the direction of signal flow in all branches of the flow graph.
2. Reverse the roles of the input and output nodes.
3. Keep the branch operations the same (delays, multiplications, etc.).

The fact that the overall transfer function stays the same is based on Mason's Rule, which is summarized at the end of this unit.

Example 1 of Transposed System

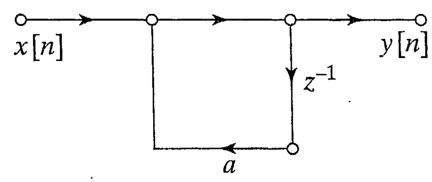
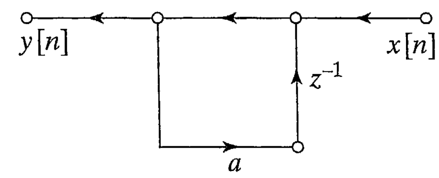
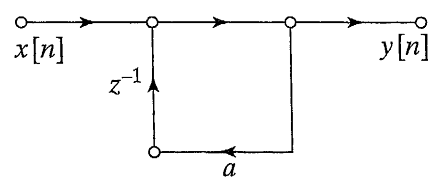


Figure 6.24 (a) Flow graph for simple 1st-order system



(b) Transposed form of (a)



(c) Structure of (b) redrawn with input on left

Example 2 of a Transposed System

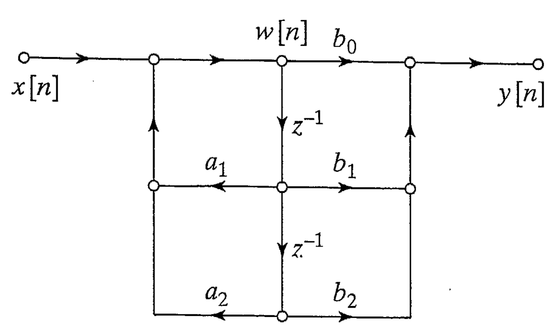


Figure 6.25 Direct Form II structure for Example 6.8

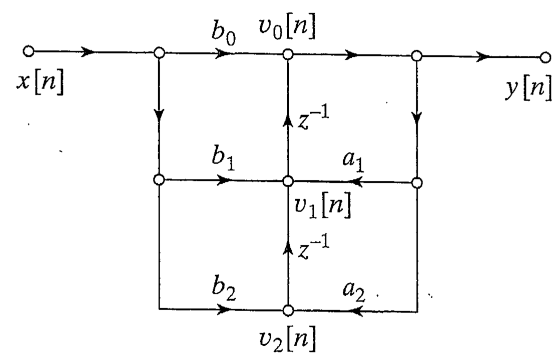


Figure 6.26 Transposed Direct Form II structure for Example 6.8

Implementation of FIR Filters

Since FIR filters can be considered a special case of IIR filters for which all the feedback coefficients are 0, the previous discussion on implementation of IIR filters also applies to the FIR cases. However, several issues should be emphasized for the FIR case:

- Direct Form I and Direct Form II are the same for FIR filters. For example, compare the following two figures for the case where all the a_k terms are 0.

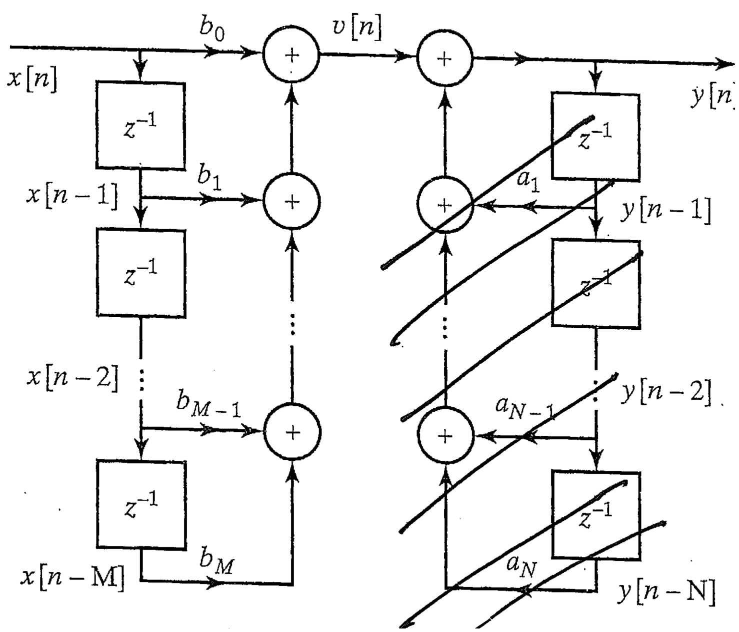


Figure 6.3 Block diagram representation for a general Nth-difference equation.

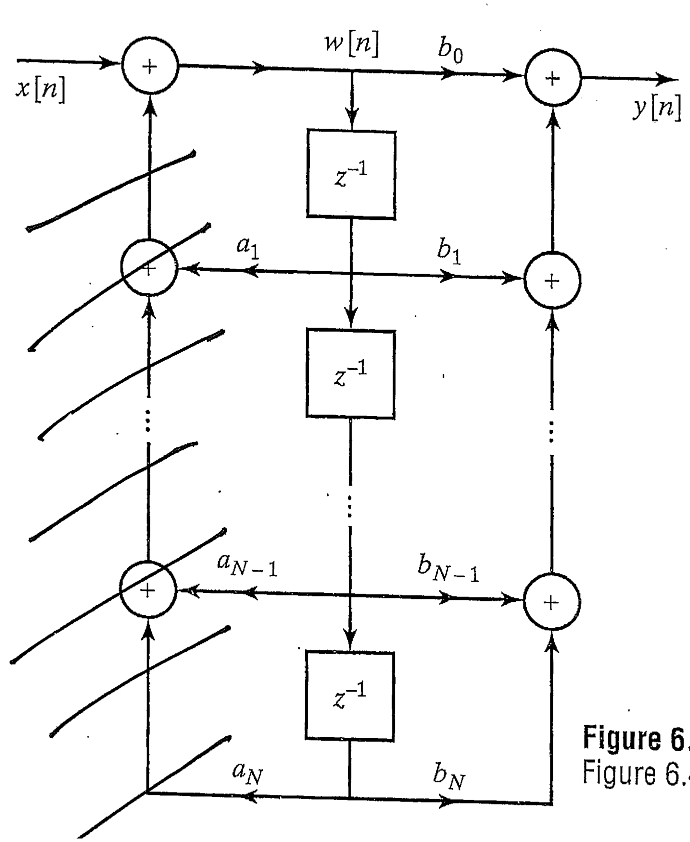


Figure 6.5 Combination of delays in Figure 6.4.

• The method of obtaining a new implementation structure by transposing an existing structure obviously still applies. See the example below of obtaining a transpose form for an FIR filter.

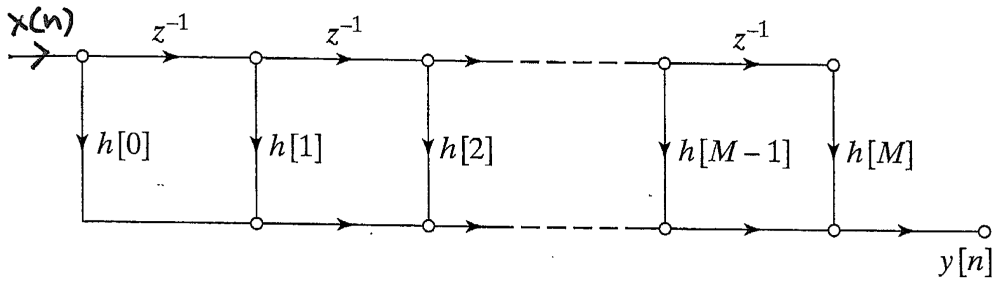


Figure 6.29 Direct-form realization of an FIR system

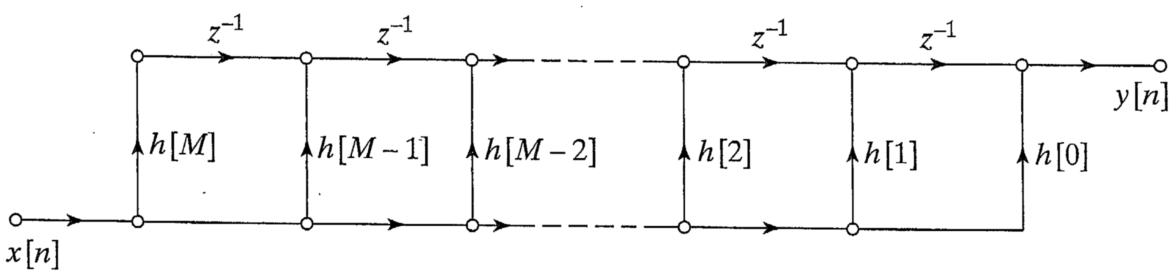


Figure 6.30 Transposition of the network of Figure 6.29.

Note: because of the structure shown in Figure 6.29, above, this structure is often called a tapped-delay line structure or a transversal filter structure.

• The <u>cascade structure</u> also applies to FIR filters. To obtain this structure, the system function is expressed as

$$H(z) = \sum_{n=0}^{M} h(n) z^{-n} = \prod_{k=1}^{M_s} (b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2}) \qquad \text{(equation 6.48)}$$

which can be implemented as shown in the figure below.

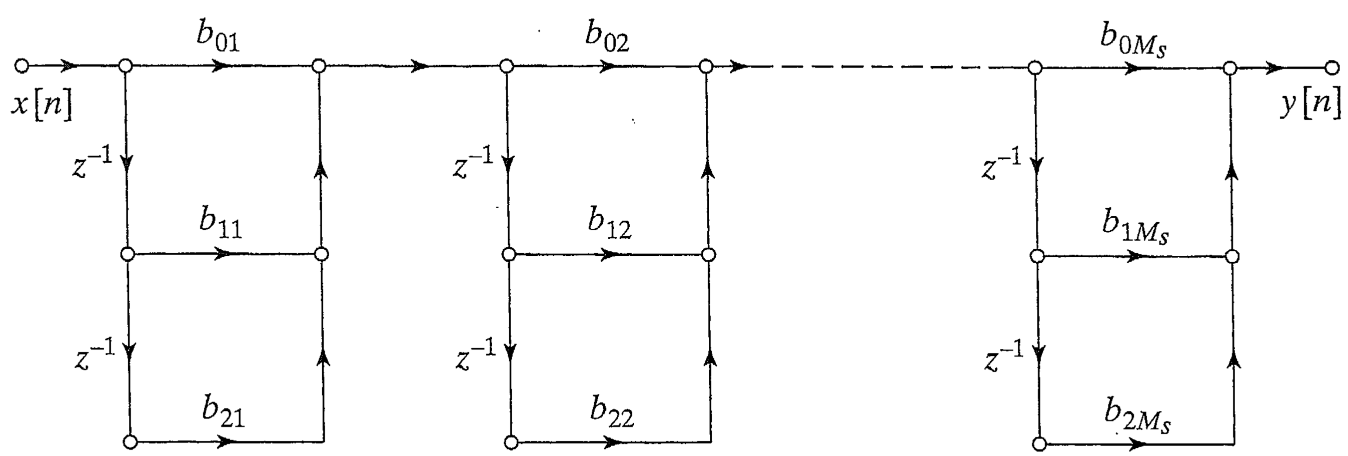


Figure 6.31 Cascade form realization of an FIR system.

Implementing FIR Filters Having Generalized Linear Phase

The symmetry property of h(n) for Types I-IV FIR filters can be used to reduce the number of multiplications per output by a factor of 2.

For Type I filters (M even, h(n) symmetric), the filter can be implemented using

$$y(n) = \sum_{k=0}^{\frac{M}{2}-1} h(k)[x(n-k) + x(n-M+k)] + h(\frac{M}{2})x(n-\frac{M}{2}). \qquad \text{(Equation 6.50)}$$

The corresponding implementation structure is shown below.

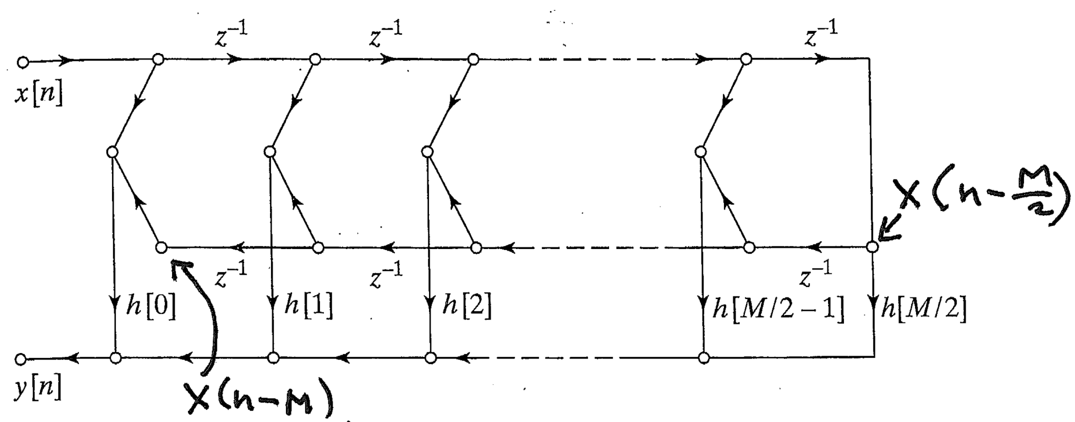


Figure 6.32 Direct form structure of an FIR linear-phase system when M is an even integer.

<u>Type II filters</u> (M odd, h(n) symmetric) can be implementing using:

$$y(n) = \sum_{k=0}^{\frac{M-1}{2}} h(k)[x(n-k) + x(n-M+k)]$$

for which the implementation structure is shown below.

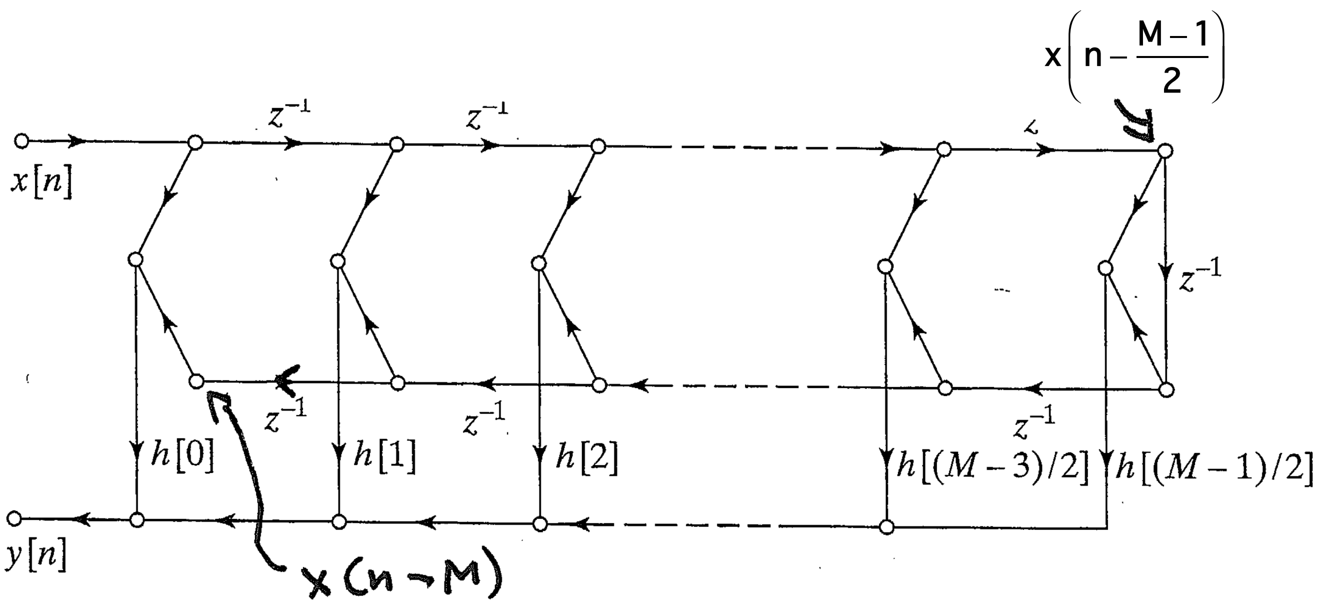


Figure 6.33 Direct form structure for an FIR linear-phase system when M is an odd integer.

For Type III filters (M even, h(n) antisymmetric) the difference equation for implementing the system is

$$y(n) = \sum_{k=0}^{\frac{M}{2}-1} h(k)[x(n-k) - x(n-M+k)]. \qquad \text{(Equation 6.51)}$$

The implementation structure is the same as one for Type I filters, except that all the x(n-j) terms traveling left on the second row in the diagram are subtracted, rather than added, to the summation node they feed. In addition, h(M/2) is always 0 for a Type III filter.

For Type IV filters, the difference equation is

$$y(n) = \sum_{k=0}^{\frac{M-1}{2}} h(k)[x(n-k) - x(n-M+k)] \qquad \text{(Equation 6.53)}$$

and the implementation structure is the same as one for Type II filters, except that all the x(n-j) terms traveling left on the second row in the diagram are subtracted, rather than added, to the summation node they feed.

<u>More on cascade implementations</u>: We have seen that the zeros of Types I-IV FIR filters occur in groups of 4, 2, or 1. One approach to implementing these filters is to use the cascade form with each multiplying factor representing 4 zeros, 2 zeros, or 1 zero according to the inherent grouping of zeros of these generalized linear phase filters. For example, consider the FIR filter having the zeros shown below:

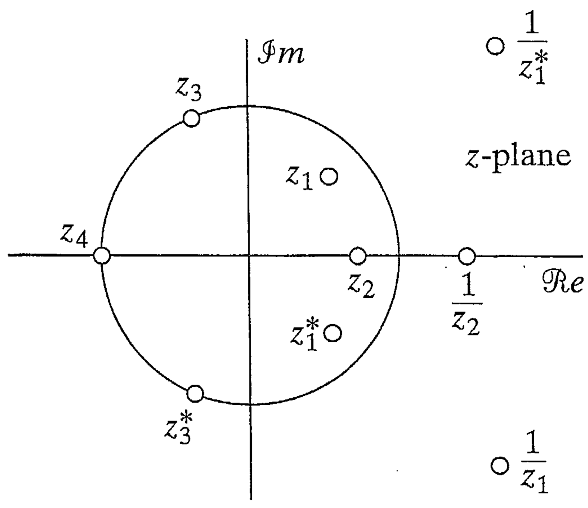


Figure 6.34 Symmetry of zeros for a linear-phase FIR filter.

This filter could be implemented in cascade form based on the following factoring of H(z):

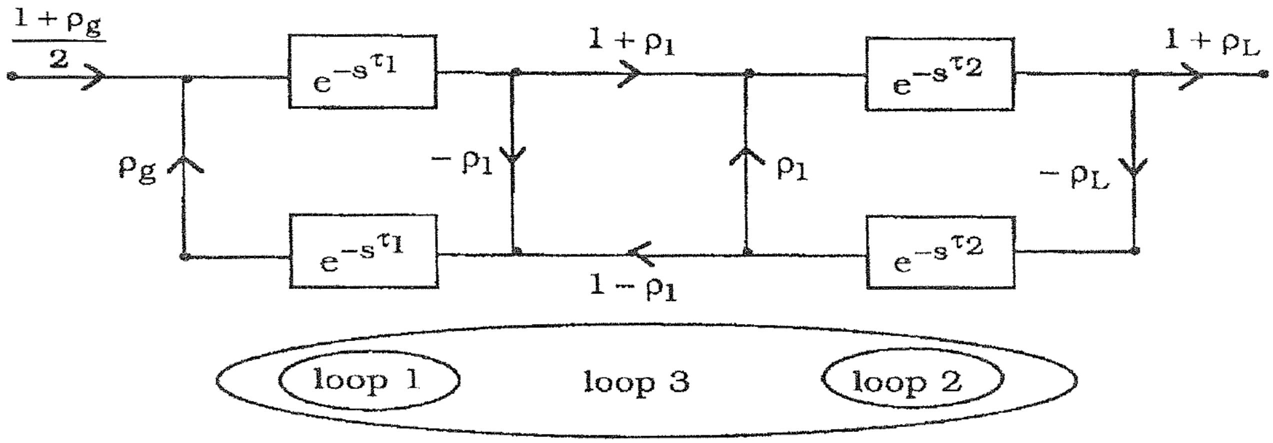
$$H(z) = h(0)(1+z^{-1})(1+az^{-1}+z^{-2})(1+bz^{-1}+z^{-2})\ (1+cz^{-1}+dz^{-2}+cz^{-3}+z^{-4})$$

where

$a = (z_2 + 1/z_2)$ $\quad$ $b = 2\text{Re}[z_3\}$ $\quad$ $c = -2\text{Re}[z_1 + 1/z_1]$ and $d = 2 + \left|z_1 + (1/z_1)\right|^2$.

Note from the symmetry of coefficients for each of the factors of H(z) indicates that each individual factor has generalized linear phase.

Application of Mason's Rule to 2-Tube Speech Production Model

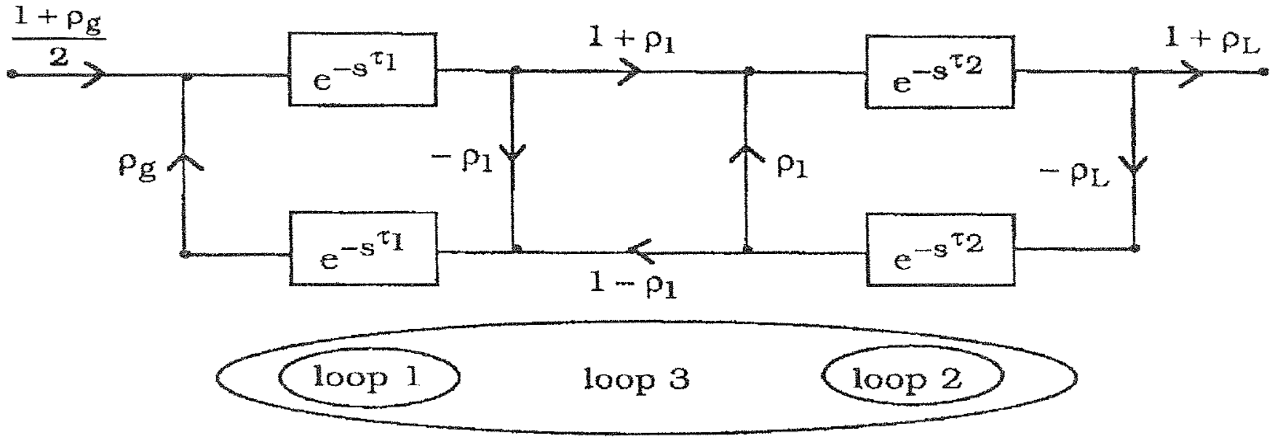


Mason's Rule:

Overall transmittance (transfer function) is given by:

$$T = \frac{\sum_n T_n \Delta_n}{\Delta}$$

where T_n = transmittance of one forward path between input and output (only 1 in this example)



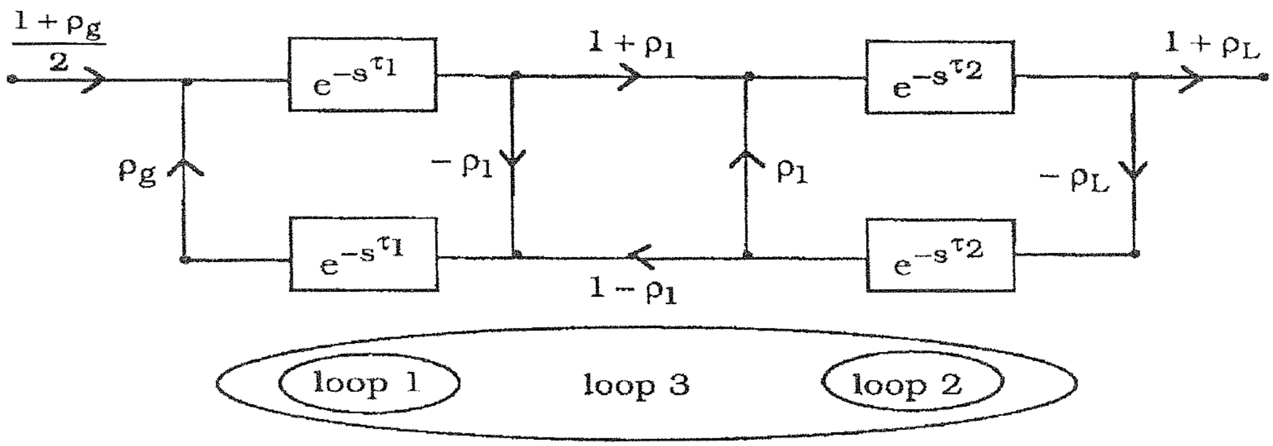
$$T_1 = \frac{1+\rho_g}{2} e^{-s\tau_1}(1+\rho_1)^{-s\tau_2}(1+\rho_L)$$

$$= \frac{1+\rho_g}{2}(1+\rho_1)(1+\rho_L)e^{-s(\tau_1+\tau_2)}$$

Also, $\Delta = 1 - \sum L_1 + \sum L_2 - \sum L_3 +$ etc.

where L_1 is the transmittance of a closed path

and $\sum L_1$ is the sum of the L_1 values for all closed paths. (which can overlap)

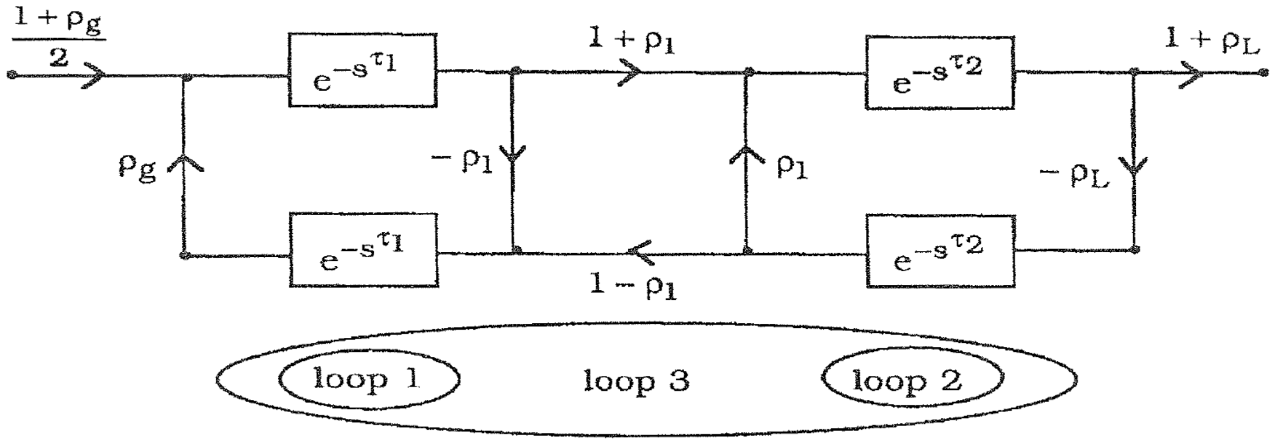


For the above example,

$$\sum L_1 = -\rho_1\rho_g e^{-s2\tau_1} - \rho_1\rho_L e^{-s2\tau_2} - \rho_g\rho_L \underbrace{(1+\rho_1)(1-\rho_1)}_{1-\rho_1^2} e^{-s(2\tau_1+2\tau_2)}$$

Also, L_2 = the product of transmittance of two non-touching loops (loops with no common nodes)

$\sum L_2$ = sum of L_2 values for all possible pairwise combinations of non-touching loops.



In the present example (2-tube) model), the only non-touching pair of loops involves Loop 1 and Loop 2.

$$\Rightarrow L_2 = \sum L_2 = \left(-\rho_1\rho_g e^{-s2\tau_1}\right)\left(-\rho_1\rho_L e^{-s2\tau_2}\right)$$

$$= \rho_1^2\rho_g\rho_L e^{-s(2\tau_1+2\tau_2)}$$

Since L_3 is the product of three non-touching loops (loops with no common nodes), $L_3 = 0$ for this example.

Therefore, for this example,

$$\Delta = 1 - \Sigma L_1 + \Sigma L_2$$

$$= 1 - \left[-\rho_1\rho_g e^{-s2\tau_1} - \rho_1\rho_L e^{-s2\tau_2} - \rho_g\rho_L(1-\rho_1^2)e^{-s(2\tau_1+2\tau_2)} \right] + \rho_1^2\rho_g\rho_L e^{-s(2\tau_1+2\tau_2)}$$
$$= 1 + \rho_1\rho_g e^{-s2\tau_1} + \rho_1\rho_L e^{-s2\tau_2} + \rho_g\rho_L e^{-s(2\tau_1+2\tau_2)}$$

Also, Δ_1 is the value of Δ for the flow graph that results when forward path T_1 is removed. In our case, there are no loops left when T_1 is removed.

$$\Rightarrow \Delta_1 = 1 - \Sigma L_1 + \Sigma L_2 = 1$$

Overall, the expression for T becomes $T = \dfrac{\sum_n T_n \Delta_n}{\Delta} = \dfrac{T_1}{\Delta}$

$$= \frac{\left(\dfrac{1+\rho_g}{2}\right)(1+\rho_1)(1+\rho_L)e^{-s(\tau_1+\tau_2)}}{1 + \rho_1\rho_g e^{-s2\tau_1} + \rho_1\rho_L e^{-s2\tau_2} + \rho_g\rho_L e^{-s(2\tau_1+2\tau_2)}} = H(s)$$

To obtain an expression for the frequency response, let $s = j\Omega$ in H(s).