IIR Filters - Direct Form Structures

An Infinite Impulse Response (IIR) filter consisting of a ratio of two polynomials can be decomposed into the cascade of an all-zero filter and an all-pole filter:

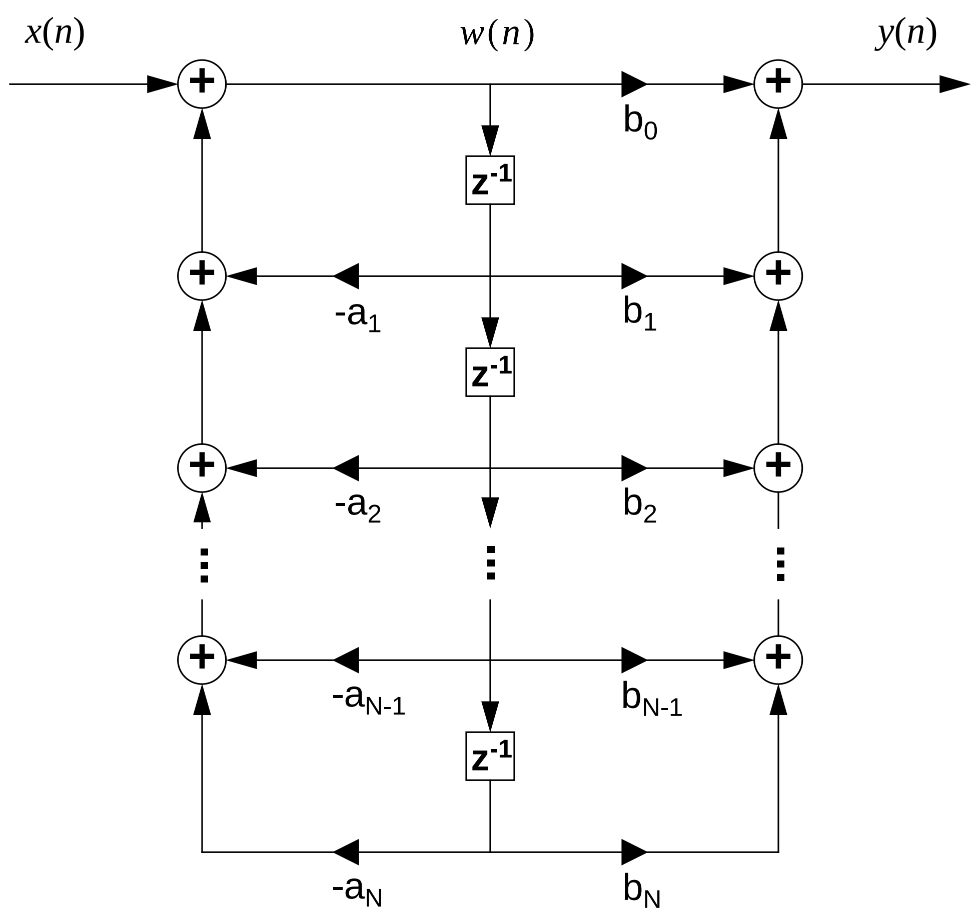
$$y(n) = -\sum_{k=1}^{N} a_k y(n-k) + \sum_{k=0}^{M} b_k x(n-k)$$

or,

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

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

This can be implemented efficiently using $(M+N+1)$ multiplications, $(M+N)$ additions, and the maximum of $\{M, N\}$ memory locations using a Direct Form II realization:



SIgnal Flow Graphs and Transposed Structures

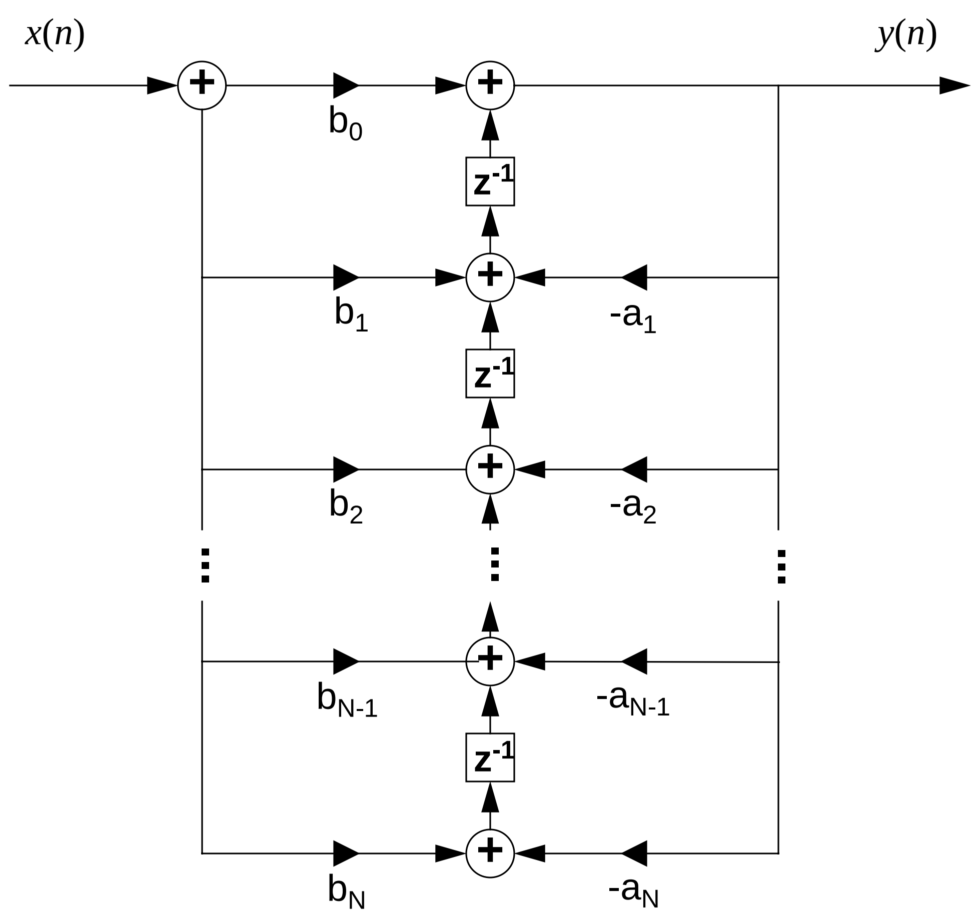
From Network Theory, we recall the following theorem:

Transposition or Flow-Graph Reversal Theorem:

> If we reverse the directions of all branch transmittances and interchange the input and output, the system function remains unchanged.

Block diagram representations of filters can be converted to signal flow graphs by treating delay elements and multipliers as weights on an arc, and replacing summers with a filled circle. This allows us to transform the problem of the design of a filter into a network topology problem (or a graph theory problem).

Extending such theory results in the following transposed structure for the Direct Form II implementation (see Table 7.1):



Cascade-Form Structures

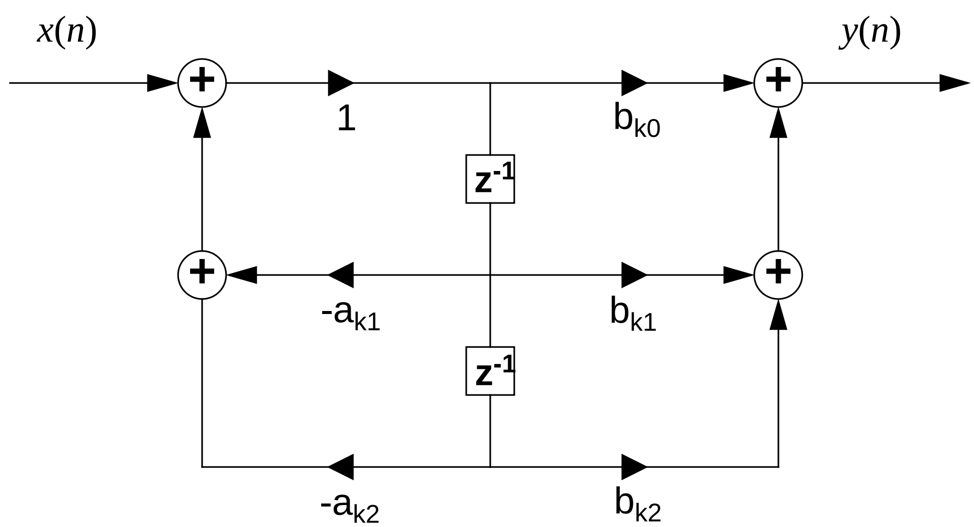
A system can be factored into a product of second-order systems:

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

where K is the integer part of $(N+1)/2$. $H_k(z)$ has the general form:

$$H_k(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

This may be implemented using the following form for each second-order section:



What are the significant differences between this realization and a direct form realization?

Parallel-Form Structures

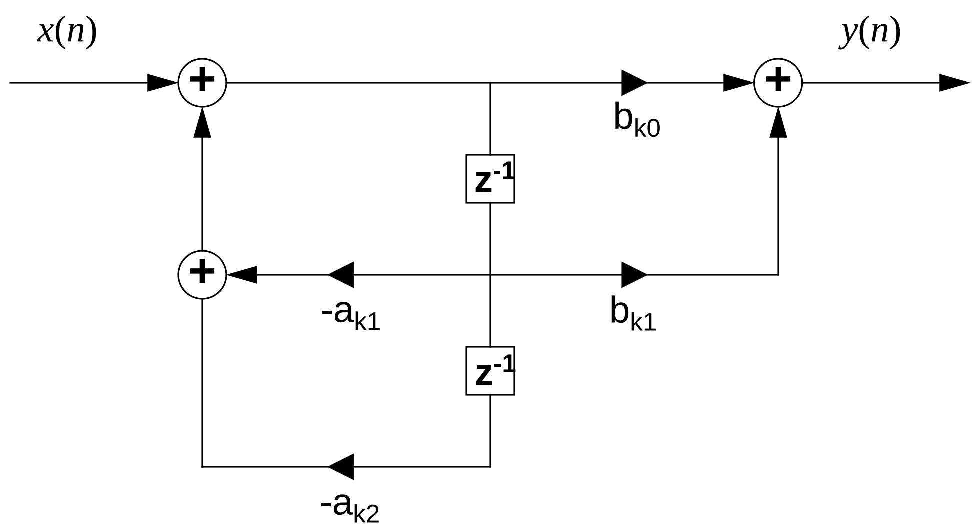
A system can be factored into a sum of second-order systems using a partial-fraction expansion:

$$H(z) = C + \prod_{k=1}^{K} H_k(z)$$

where K is the integer part of $(N+1)/2$. $H_k(z)$ has the general form:

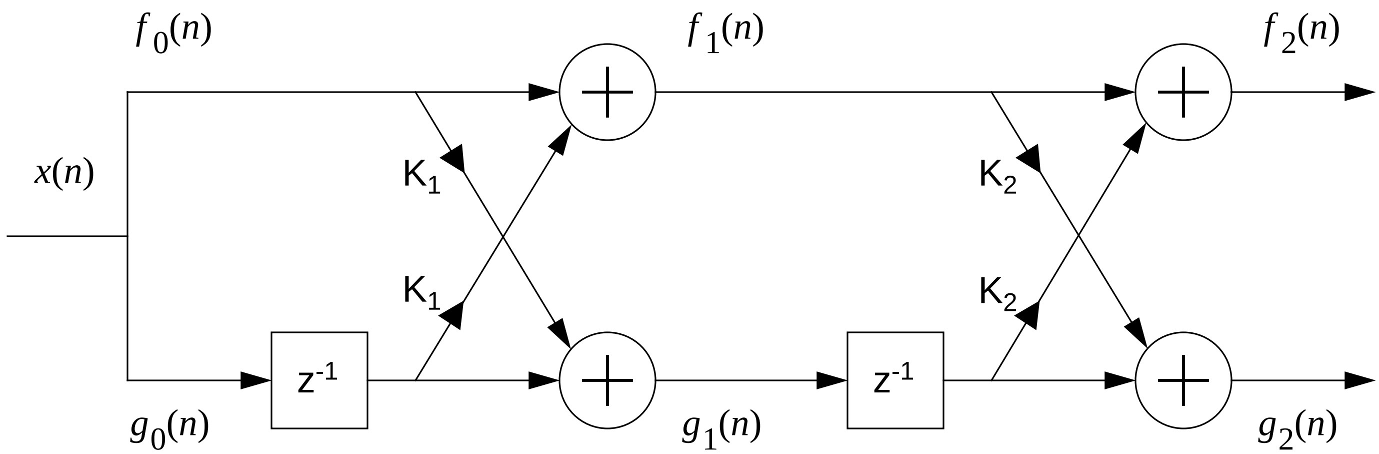
$$H_k(z) = \frac{b_{k0} + b_{k1}z^{-1}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

This may be implemented using the following form for each second-order section:

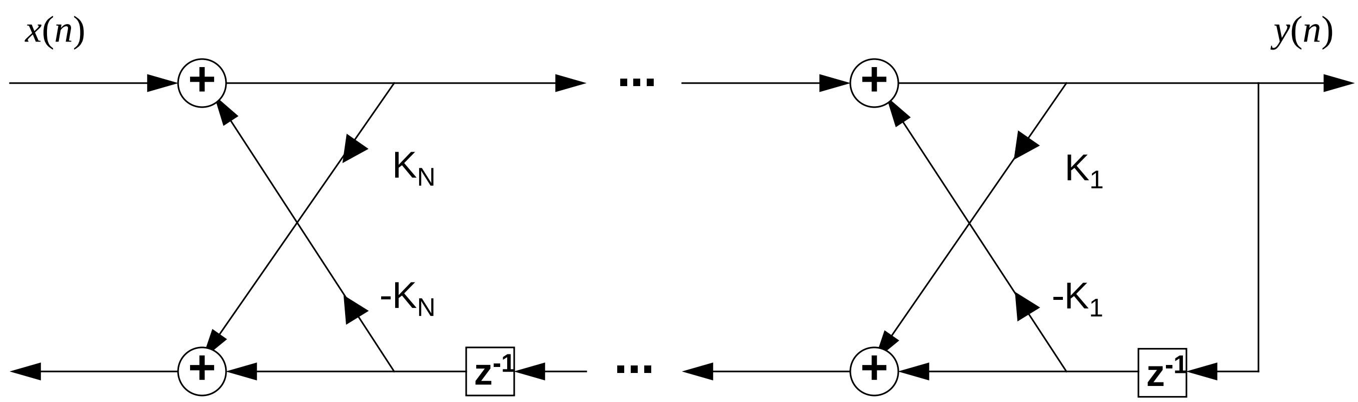


Lattice and Ladder Structures

For the FIR lattice structure:



the inverse, or IIR equivalent, is:

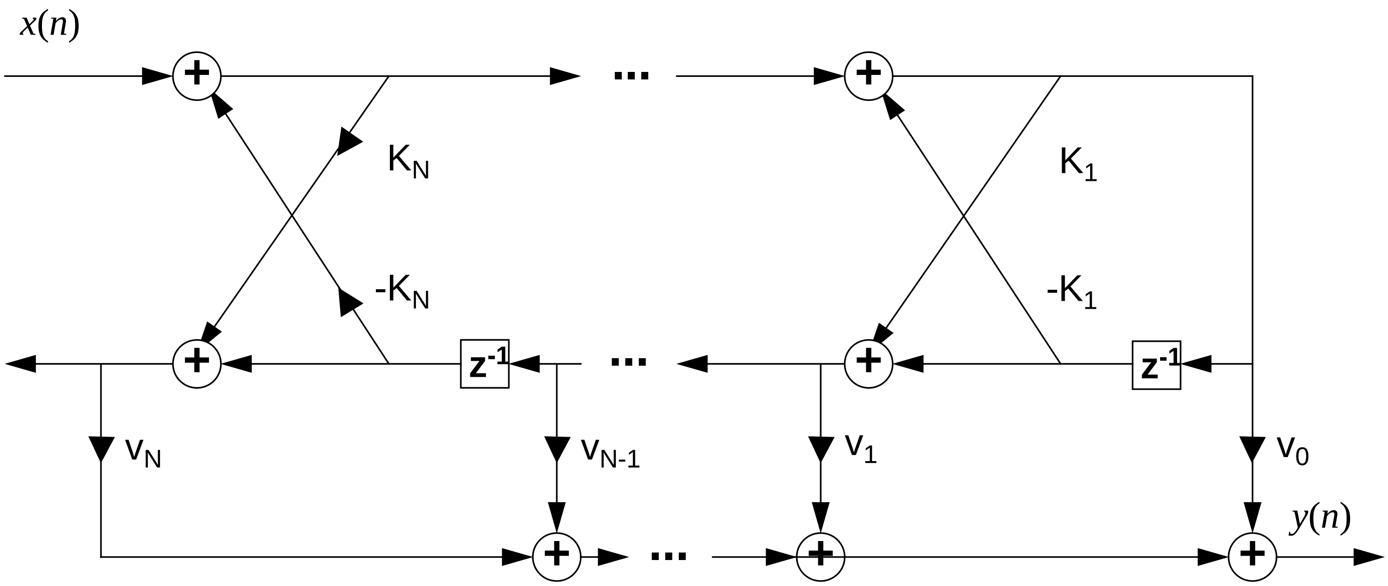


This structure can be generalized to implement a pole/zero filter. Consider the filter:

$$H(z) = \frac{\sum_{k=0}^{M} c_M(k) z^{-k}}{1 + \sum_{k=1}^{N} a_N(k) z^{-k}} = \frac{C_M(z)}{A_N(z)}$$

We already know that $A_N(z)$ can be implemented with the above lattice filter.

Consider the lattice-ladder filter shown below:



The coefficients of the ladder filter, $\{v_k\}$, can be found from the following recursion:

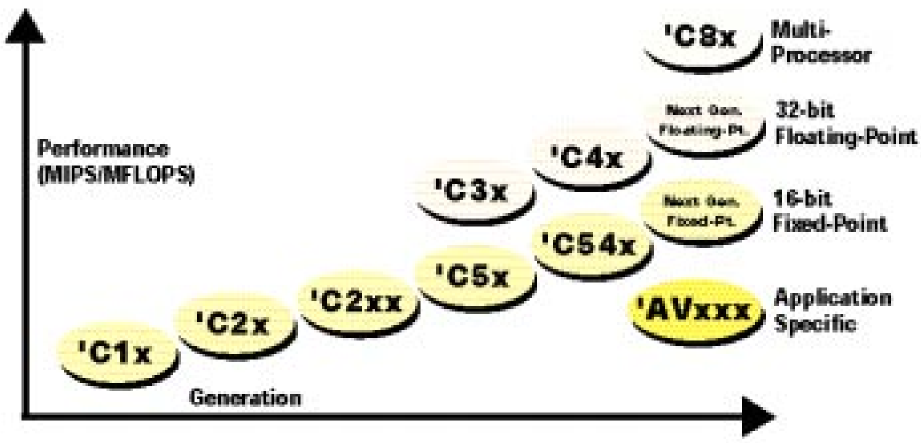
$$v_m = c_m(m)$$
$$C_{m-1}(z) = C_m(z) - v_m B_m(z)$$

The recursion is performed backwards: m=M,M-1,...,2.

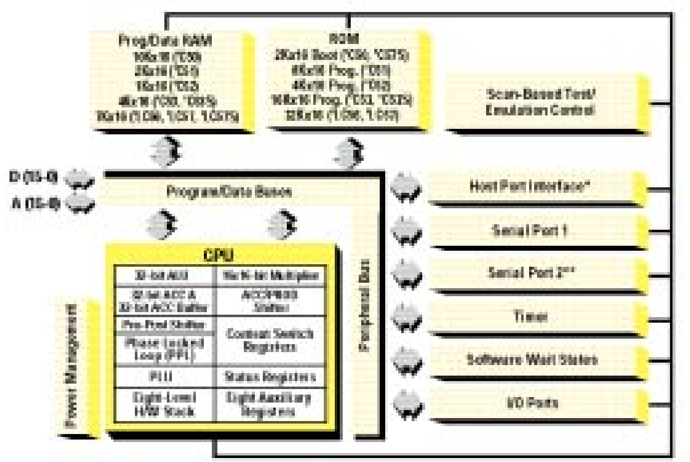
At each stage of the iteration, v_m is computed, and then $C_{m-1}(z)$ is computed from v_m and $B_m(z)$. Next, $B_{m-1}(z)$ is computed using the step-down procedure previously described.

Ladder filters find applications in channel equalization (such as modems).

Software Implementation of Discrete-Time Systems



'C5x Block Diagram



* Available on the 'LC57 and 'C57S. ** One serial port on 'C52; buffered serial port on 'LC56, 'LC57 and 'C57S

'C8x Architecture