

FILTERS SUMMARY

realization of digital filters direct form I direct form II

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 or ASIC 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.

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:

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1z^{-1} + b_2z^{-2} + \dots + b_Nz^{-N}}{1 + a_1z^{-1} + a_2z^{-2} + \dots + a_Mz^{-M}}$$

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.

Analysis techniques

A variety of mathematical techniques may be employed to analyze the behaviour of a given digital filter. Many of these analysis techniques may also be employed in designs, and often form the basis of a filter specification.

Typically, one characterizes filters by calculating how they will respond to a simple input such as an impulse. One can then extend this information to compute the filter's response to more complex signals.

Impulse response

The impulse response, often denoted $h[k]$ or h_k , is a measurement of how a filter will respond to the Kronecker delta function. For example, given a difference equation, one would set $x_0 = 1$ and $x_k = 0$ for $k \neq 0$ and evaluate. The impulse response is a characterization of the filter's behaviour. 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:

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

IIR filters on the other hand are recursive, with the output depending on both current and previous inputs as well as previous outputs. The general form of an IIR filter is thus:

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

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

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:

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

This is expanded:

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

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, $y[n]$, in terms of the past outputs, $y[n - p]$, the present input, $x[n]$, and the past inputs, $x[n - p]$. 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 evaluation.

Filter design[edit]

The design of digital filters is a deceptively complex topic. Although filters are easily understood and calculated, the practical challenges of their design and implementation are significant and are the subject of much advanced research.

There are two categories of digital filter: the recursive filter and the nonrecursive filter. These are often referred to as infinite impulse response (IIR) filters and finite impulse response (FIR) filters, respectively

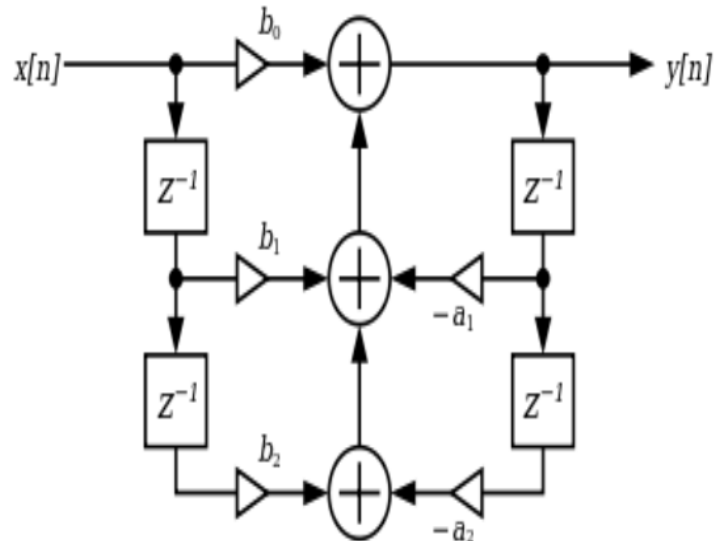
Filter realization

After 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.

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. In general, this form requires $2N$ delay elements (for both input and output signals) for a filter of order N .



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

