CHAPTER 10.6
DIGITAL FILTERS*

Arthur B. Williams, Fred J. Taylor

RELATION TO ANALOG FILTERS

Digital filters provide many of the same frequency selective services (low, high, bandpass) expected of analog filters. In some cases, digital filters are defined in terms of equivalent analog filters. Other digital filters are designed using rules unique to this technology. The hardware required to fabricate a digital filter are basic digital devices such as memory and arithmetic logic units (ALUs). Many of these hardware building blocks provide both high performance and low cost. Coefficients and data are stored as digital computer words and, as a result, provide a precise and noise free (in an analog sense) signal processing medium. Compared to analog filters, digital filters generally enjoy the following advantages:

1. They can be fabricated in high-performance general-purpose digital hardware or with application-specific integrated circuits (ASIC).
2. The stability of certain classes of digital filters can be guaranteed.
3. There are no input or output impedance matching problems.
4. Coefficients can be easily programmed and altered.
5. Digital filters can operate over a wide range of frequencies.
6. Digital filters can operate over a wide dynamic range and with high precision.
7. Some digital filters provide excellent phase linearity.
8. Digital filters do not require periodic alignment and do not drift or degrade because of aging.

DATA REPRESENTATION

In an analog filter, all signals and coefficients are considered to be real or complex numbers. As such, they are defined over an infinite range with infinite precision. In the analog case, filter coefficients are implemented with lumped R, L, C, and amplifier components of assumed absolute precision. Along similar lines, the designs of digital filters typically begin with the manipulation of idealized equations. However, the implementation of a digital filter is accomplished using digital computational elements of finite precision (measured in bits). Therefore, the analysis of a digital filter is not complete until the effects of finite precision arithmetic has been

*This section is based on the author's *Electronic Filter Design Handbook*, 3rd ed., McGraw-Hill, 1995.

determined. As a result, even though there has been a significant sharing of techniques in the area of filter synthesis between analog and digital filters, the analyses of these two classes of filters have developed separate tools and techniques.

Data, in a digital system, are represented as a set of binary-valued digits. The process by which a real signal or number is converted into a digital word is called analog-to-digital conversion (ADC). The most common formats used to represent data are called *fixed* and *floating* point (FXP and FLP). Within the fixed-point family of codes, the most popular are binary-coded decimal sign magnitude (SM) and diminished radix (DR) codes. Any integer X such that $|X| < 2^{n-1}$ has a unique sign-magnitude representation, given by

$$X = X_{n-1}\text{:}\ (2^{n-2} X_{n-2} + \cdots + 2X_1 + X_0) \tag{1}$$

where X_i is the ith bit and X_0 is referred to as the least significant bit (LSB). Similarly X_{n-2} is called the most significant bit (MSB) and X_{n-1} is the sign bit. The LSB often corresponds to a physically measurable electrical unit. For example, if a signed 12-bit ADC is used to digitize a signal whose range is ±15 V, the LSB represents a quantization step size of $Q = 30$ V (range)/2^{12} – bits = 7.32 mV/bit.

Fractional numbers are also possible simply by scaling X by a power of 2. The value of $X' = X/2^m$ has the same binary representation of X except that the m LSBs are considered to be fractional bits.

SIGNAL REPRESENTATION

An analog filter manipulates real signals of assumed infinite precision. In a discrete system, analog signals of assumed infinite precision are periodically sampled at a rate of f samples per second. The same period is therefore given by $t_s = 1/f_s$ second(s). A string of contiguous samples is called a *time series*. If the samples are further processed by an ADC, a digital time series results. A digital filter can be used to manipulate this time series using digital technology. The hardware required to implement such a filter is the product of the microelectronics revolution.

SPECTRAL REPRESENTATION

Besides representing signals in the continuous or discrete time domain, signals can also be modeled in the frequency domain. This condition is called *spectral representation*. The principal tools used to describe a signal in the frequency domain are: (1) Fourier transforms, (2) Fourier series, and (3) discrete Fourier transforms (DFT).

A Fourier transform will map an arbitrary transformable signal into a continuous frequency spectrum consisting of all frequency components from –∞ to +∞. The Fourier transform is defined by an indefinite integral equation whose limits range from –∞ to +∞. The Fourier series will map a continuous but periodic signal of period T [i.e., $x(t) = x(t + kT)$ for all integer values of k] into a discrete but infinite spectrum consisting of frequency harmonics located at multiples of the fundamental frequency $1/T$. The Fourier series is defined by a definite integral equation whose limits are $[0, T]$. The discrete Fourier transform differs from the first two transforms in that it does not accept data continuously but rather from a time series of finite length. Also, unlike the first two transforms, which produce spectra ranging out to ±∞ Hz, the DFT spectrum consists of a finite number of harmonics.

The DFT is an important and useful tool in the study of digital filters. The DFT can be used to both analyze and design digital filters. One of its principal applications is the analysis of a filter's impulse response. An impulse response database can be directly generated by presenting a one-sample unit pulse to a digital filter that is initially at a zero state (i.e., zero initial conditions). The output is the filter's impulse response, which is observed for N contiguous samples. The N-sample database is then presented to an N-point DFT, transformed, and analyzed. The spectrum produced by the DFT should be a reasonable facsimile of the frequency response of the digital filter under test.

FILTER REPRESENTATION

A transfer function is defined by the ratio of output and input transforms. For digital filters, it is given by $H(z) = Y(z)/U(z)$ where $U(z)$ is the z transform of the input signal $u(n)$ and $Y(z)$ is for the output signal $y(n)$. The frequency response of a filter $H(z)$ can be computed using a DFT of the filter's impulse response.

Another transform tool that also is extensively used to study digital filters is the *bilinear z transform*. While the standard z transform can be related to the simple sample and hold circuit, the bilinear z transform is analogous to a first-order hold. The bilinear z transform is related to the familiar Laplace transform through

$$s = \frac{2(z-1)}{t_s(z+1)} \qquad z = \frac{(2/t_s)+s}{(2/t_s)-s} \tag{2}$$

Once an analog filter $H(s)$ is defined, it can be converted into a discrete filter $H(z)$ by using the variable substitution rule.

FINITE IMPULSE-RESPONSE (FIR) FILTERS

Linear constant coefficient filters can be categorized into two broad classes known as finite impulse-response (FIR) or infinite impulse-response (IIR) filters. An FIR filter can be expressed in terms of a simple discrete equation:

$$y(n) = c_0x(n) + c_1x(n-1) + \cdots + c_{N-1}x(n-N+1) \tag{3}$$

where the coefficients $\{C_i\}$ are called *filter tap weights*. In terms of a transfer function, Eq. (3) can be restated as

$$H(z) = \sum_{i=0}^{n-1} C_i z^{-1} \tag{4}$$

As an example, a typical $N = 111$th-order FIR is shown in Fig. 10.6.1. The FIR exhibits several interesting features:

1. The filter's impulse response exists for only $N = 111$ (finite) contiguous samples.
2. The filter's transform function consists of zeros only (i.e., no poles). As a result, an FIR is sometimes referred to as an all-zero, or transversal, filter.
3. The filter has a very simple design consisting of a set of word-wide shift registers, tap-weight multipliers, and adders (accumulators).
4. If the input is bounded by united (i.e., $|x(i)| \leq 1$ for all i), the maximum value of the output $y(i)$ is $\Sigma|C_i|$. If all the tap weights C_i are bounded, the filter's output is likewise bounded and, as a result, stability is guaranteed.
5. The phase, when plotted with respect to frequency (plot shown over the principal angles ± π/2), is linear with constant slope.

LINEAR PHASE BEHAVIOR

The FIR is basically a shift-register network. Since digital shift registers are precise and easily controlled, the FIR can offer the designer several phase domain attributes that are difficult to achieve with analog filters. The most important of these are: (1) Potential for linear phase versus frequency behavior and (2) potential for constant

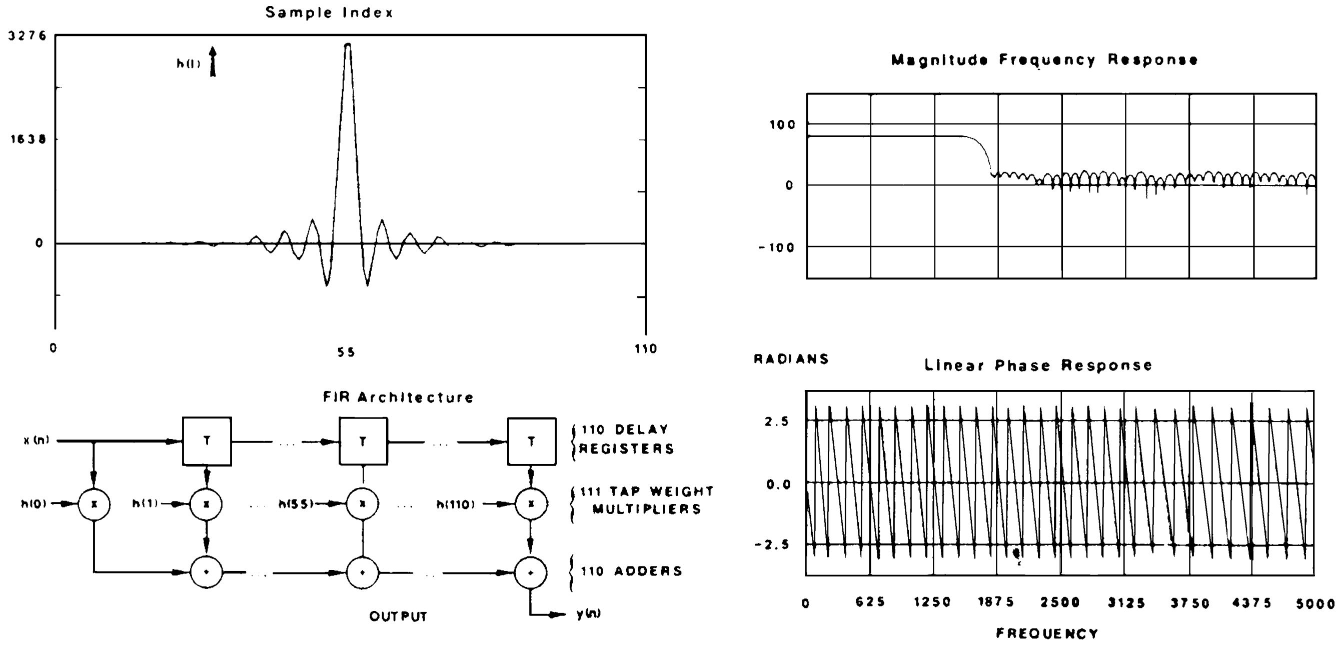

FIGURE 10.6.1 Typical FIR architecture, impulse response, and frequency response.

group-delay behavior. These properties are fundamentally important in the fields of digital communications systems, phase synchronization systems (e.g., phase-locked loops), speech processing, image processing, spectral analysis (e.g., Fourier analysis), and other areas where nonlinear phase distortion cannot be tolerated.

FIR DESIGN METHODS

The design of an FIR entails specifying the filter's impulse response, tap weights $\{C_i\}$. As a result, the design of an FIR can be as simple as prespecifying the desired impulse response. Other acceptable analytical techniques used to synthesize a desired impulse response are the inverse Fourier transform of a given frequency domain filter specification or the use of polynomial approximation techniques. These methods are summarized below.

A simple procedure for designing an FIR is to specify an acceptable frequency domain model, invert the filter's spectral representation using the inverse Fourier transform, and use the resulting time series to represent the filter's impulse response. In general, the inverse Fourier transform of a *desired* spectral waveshape would produce an infinitely long time domain record. However, from a hardware cost or throughput standpoint, it is unreasonable to consider the implementing of an infinitely or extremely long FIR. Therefore, a realizable FIR would be defined in terms of a truncated Fourier series. For example, the Fourier transform of the "nearly ideal" $N = 101$ order low-pass filter has a sin $(x)/x$ type impulse-response envelope. For a large value of N, the difference between the response of an infinitely long impulse response and its N-sample approximation is small; however, when N is small, large approximation errors can occur.

Optimal Modeling Techniques

Weighted Chebyshev polynomials have been successfully used to design FIRs. In this application, Chebyshev polynomials are combined so that their combined sum minimizes the maximum difference between an ideal and the realized frequency response (i.e., mini-max principle). Because of the nature of these polynomials, they produce a "rippled" magnitude frequency-response envelope of equal minima and maxima in the pass- and stopbands. As a result, this class of filters is often called an *equiripple* filter. Much is known about the synthesis process, which can be traced back to McClellan et al.[48] Based on these techniques, a number of software-based CAD tools have been developed to support FIR design.

WINDOWS

Digital filters usually are expected to operate over long, constantly changing data records. An FIR, while being capable of offering this service, can only work with a limited number of samples at a time. A similar situation presents itself in the context of a discrete Fourier transform. The quality of the produced spectrum is a function of the number of transformed samples. Ideally, an infinitely long impulse response would be defined by an ideal filter. A *uniform window* of length T will pass N contiguous samples of data. The windowing effect may be modeled as a multiplicative switch that multiplies the presented signal by zero (open) for all time exclusive of the interval $[0, T]$. Over $[0, T]$, the signal is multiplied by unity (closed). In a sampled system, the interval $[0, T]$ is replaced by N samples taken at a sample rate f_s where $T = N/f_s$. When the observation interval (i.e., N) becomes small, the quality of the spectral estimate begins to deteriorate. This consequence is called the *finite aperture effect*.

Windowing is a technique that tends to improve the quality of a spectrum obtained from a limited number of samples. Some of the more popular windows found in contemporary use are the rectangular or uniform window, the Hamming window, the Hann window, the Blackman window, and the Kaiser window.

Windows can be directly applied to FIRs. To window an N-point FIR, simply multiply the tap weight coefficients C_i with the corresponding window weights w_i. Note that all of the standard window functions have even symmetry about the midsample. As a result, the application of such a window will not disturb the linear phase behavior of the original FIR.

MULTIRATE SIGNAL PROCESSING

Digital signal processing systems accept an input time series and produce an output time series. In between, a signal can be modified in terms of its time and/or frequency domain attributes. One of the important functions that a digital signal processing system can serve is that of sample rate conversion. As the name implies, a sample rate converter changes a system's sample rate from a value of f_{in} samples per second to a rate of f_{out} samples per second. Such devices are also called multirate systems since they are defined in terms of two or more sample rates. If $f_{\text{in}} > f_{\text{out}}$ then the system is said to perform decimation and is said to be decimated by an integer M if

$$M = \frac{f_{\text{out}}}{f_{\text{in}}} \tag{5}$$

In this case, the decimated time series $x_d[n] = x[Mn]$, or every Mth sample of the original time series is retained. Furthermore, the effective sample rate is reduced from f_{in} to $f_{\text{dec}} = f_{\text{in}}/M$ samples per second.

Applications of decimation include audio and image signal processing involving two or more subsystems having dissimilar sample rates. Other applications occur when a high data rate ADC is placed at the front end of a system and the output is to be processed parameters that are sampled at a very low rate by a general-purpose digital computer. At other times, multirate systems are used simply to reduce the Nyquist rate to facilitate computational intensive algorithms, such as a digital Fourier analyzer, to be performed at a slower arithmetic rate. Another class of applications involves processing signals, sampled at a high data rate, through a limited bandwidth channel.

QUADRATURE MIRROR FILTERS (QMF)

We have stated that multirate systems are often used to reduce the sample rate to a value that can be passed through a band-limited communication channel. Supposedly, the signal can be reconstructed on the receiver side. The amount of allowable decimation has been established by the Nyquist sampling theorem. When the bandwidth of the signal establishes a Nyquist frequency that exceeds the bandwidth of a communication channel, the signal must be decomposed into subbands that can be individually transmitted across band-limited channels. This technique uses a bank of band-limited filters to break the signal down into a collection of subbands that fit within the available channel bandwidths.

Quadrature mirror filters (QMF) are often used in the subband application described in Fig. 10.6.2. The basic architecture shown in that figure defines a QMF system and establishes two input-output paths that have a bandwidth requirement that is half the input or output requirements. Using this technique, the channels can be subdivided over and over, reducing the bandwidth by a factor of 2 each time. The top path consists of low-pass filters and the bottom path is formed by high-pass filters.

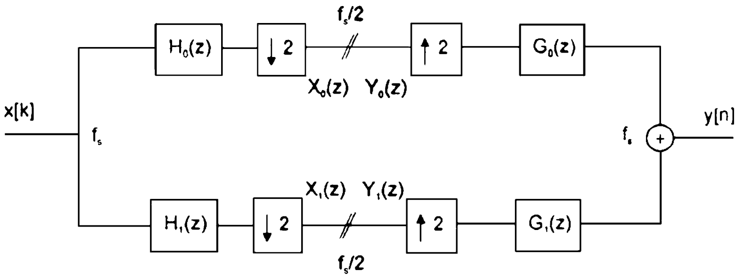

FIGURE 10.6.2 Quadrature mirror filter (QMF).

Designing QMF is, unfortunately, not a trivial process. No meaningful flat response linear phase QMF filter exists. Most QMF designs represent some compromise.

INFINITE IMPULSE-RESPONSE FILTER

The FIR filter exhibits superb linear phase behavior; however, in order to achieve a high-quality (steep-skirt) magnitude frequency response, a high-order FIR is required. Compared to the FIR, the IIR filter

1. Generally satisfies a given magnitude frequency-response design objective with a lower-order filter.
2. Does not generally exhibit linear phase or constant group-delay behavior.

If the principal objective of the digital filter design is to satisfy the prespecified magnitude frequency response, an IIR is usually the design of choice. Since the order of an IIR is usually significantly less than that of an FIR, the IIR would require fewer coefficients. This translates into a reduced multiplication budget and an attendant saving in hardware and cost. Since multiplication is time consuming, a reduced multiplication budget also translates into potentially higher sample rates.

From a practical viewpoint, a realizable filter must produce bounded outputs if stimulated by bounded inputs. The magnitude is bounded on an IIR's impulse response, namely,

$$\sum_{n=0}^{\infty} |h(n)| < M \tag{6}$$

If M is finite (bounded), the filter is stable, and if it is infinite (unbounded), the filter is unstable. This condition can also be more conveniently related to the pole locations of the filter under study. It is well known that a causal discrete system with a rational transfer function $H(z)$ is stable (i.e., bounded inputs produce bounded outputs) if and only if its poles are interior to the unit circle in the z domain. This is often referred to as the circle criterion and it can be tested using general-purpose computer root-finding methods. Other algebraic tests—Schur-Cohen, Routh-Hurwitz, and Nyquist—may also be used. The stability condition is implicit to the FIR as long as all N coefficients are finite. Here the finite sum of real bounded coefficients will always be bounded.

DESIGN OBJECTIVES

The design of an IIR begins with a magnitude frequency-response specification of the target filter. The filter's magnitude frequency response is specified as it would be for an analog filter design. In particular, assume that a filter with a magnitude-squared frequency response given by $|H(\omega)|^2$, having passband, transition band, and stopband behavior as suggested by Fig. 10.6.3, is to be synthesized. The frequency response of the desired filter is specified in terms of a cutoff critical frequency ω_p a stopband critical frequency ω_s and stop- and passband delimiters ϵ and A. The critical frequencies ω_p and ω_a represent the end of the passband and the start of the stopband, respectively, for the low-pass example. In decibels, the gains at these critical frequencies are given by (passband ripple constraint) and $-A_a = -10 \log (A^2)$ (stopband attenuation). For the case where $\epsilon = 1$, the common 3-dB passband filter is realized.

FIR AND IIR FILTERS COMPARED

The principal attributes of FIR are its simplicity, phase linearity, and ability to decimate a signal. The strength of the IIR is its ability to achieve high-quality (steep-skirt) filtering with a limited number order design. Those positive characteristics of the FIR are absent in the IIR and vice versa.

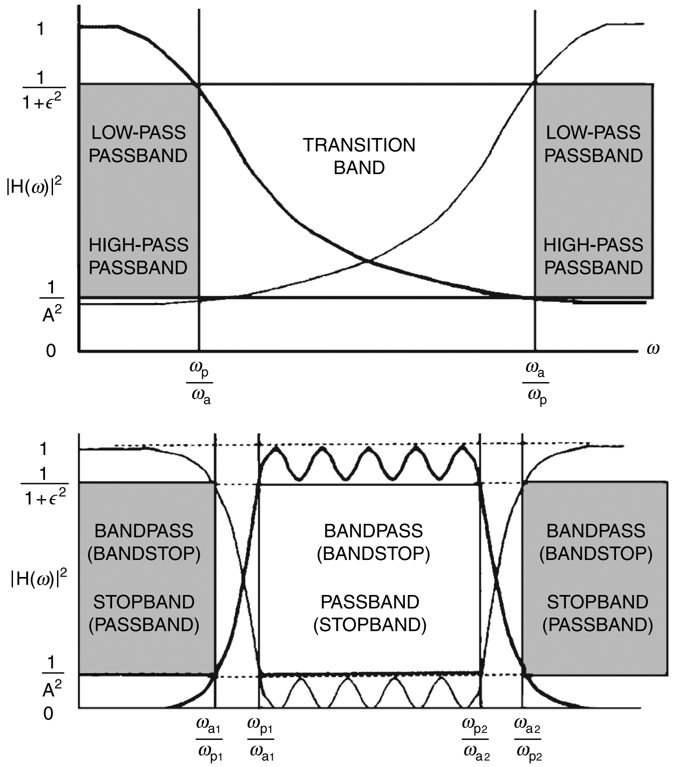

FIGURE 10.6.3 Typical design objective for low-pass, high-pass, and bandstop IIR filters.

The estimated order of an FIR, required to achieve an IIR design specification, was empirically determined by Rabiner. It was found that the approximate order n of an FIR required to meet a design objective

$$\begin{gathered}(1-\delta_1)^2 = \frac{1}{1+\varepsilon^2} \quad \delta_1 \quad \text{(passband-ripple)} \\ \delta_2^2 = \frac{1}{A^2} \quad \delta_2 \text{ (stopband bound)} \\ \Delta f \ \text{(transition frequency range}/f_s) \end{gathered} \tag{7}$$

is given by

$$n \sim \frac{-10\log((1-\delta_1)\delta_2)-15}{14\Delta f}+1 \tag{8}$$

STANDARD FILTER FORMS

The standard filter forms found in common use are: (1) Direct II, (2) Standard, (3) Cascade, and (4) Parallel. These basic filter cases are graphically interpreted in Ref. 52. The direct II and standard architectures are somewhat similar in their structure. Both strategies possess information feedback paths ranging from one delay to *n* delays. The transfer function denominator is an *n*th-order polynomial. The cascade and parallel models are constructed using a system of low-order subsections or subfilters. In the cascade design, the low-order subsections are serially interconnected. In the parallel filter, these sections are simply connected in parallel. The low-order subfilters, in both cases, are the result of factoring the *n*th-order transfer function polynomial into lower-order polynomials.

The design and analysis of all four classes of filters can be performed using manually manipulated equations or a digital computer. The most efficient method of formulating the filter design problem, whether using tables, calculators, or a computer, is called the *state-variable* technique. A state variable is a parameter that represents the information stored in a system. The set of state variables is called a *state vector*. For an analog system, information is stored on capacitors or in inductors. In earlier chapters, state variables were used to specify and facilitate the manipulation of the *R*, *L*, and *C* components of an analog filter. In these cases, capacitive voltage and inductive current were valid state variables. Since resistors have no memory, they would not be the source of a state variable.

In digital filters, the memory element, which stores the state information, is simply a delay (shift) register. The realization of digital filters is described in Ref. 52.

FIXED-POINT DESIGN

An IIR, once designed and architected, often needs to be implemented in hardware. The choices are fixed- or floating-point. Of the two, fixed-point solutions generally provide the highest real-time bandwidth at the lowest cost. Unfortunately, fixed-point designs also introduce errors that are not found in more expensive floating-point IIR designs. The fixed-point error sources are either low-order inaccuracies, because of finite precision arithmetic and data (coefficient) roundoff effects or potentially large errors because of run-time dynamic range overflow (saturation).

Additional precision can be gained by increasing the number of fractional bits assigned to the data and coefficients fields with an attendant decrease in dynamic range and an increased potential for runtime overflow. On the other hand, the overflow saturation problem can be reduced by enlarging the dynamic range of the system by increasing the integer bit field with an accompanying loss of precision. The problem facing the fixed-point filter design, therefore, is achieving a balance between the competing desire to maximize precision and to simultaneously eliminate (or reduce) run-time overflow errors. This is called the *binary-point assignment problem*.