

WIRELESS SYSTEMS DESIGN

**Master
REFERENCE**

Multirate Filter Design Makes 3G Wireless Systems More Efficient

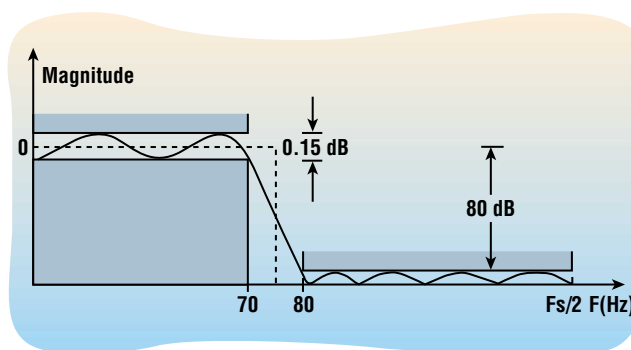
BY DON OROFINO, PAUL PACHECO, AND NICK ADLER

ONE significant challenge for developers of third-generation (3G) wireless systems is the design of digital filters for receiver processing. In order to transmit increasing amounts of information within a finite frequency bandwidth, tighter filter specifications often arise. For example, in order to mitigate adjacent-channel interference, filters with narrow passbands and transition bands are required. These filter specifications typically require higher-order filters, implying increased storage for the filter coefficients and higher execution rates for the processor. Moreover, high-order filters can be difficult, if not impossible, to design.

Multirate filters are a practical solution to design and implement finite-impulse-response (FIR) filters with narrow spectral constraints. Multirate filters change the input data rate at one or more intermediate points within the filter itself, while maintaining an output rate that is identical to the input rate. These filters achieve greatly reduced filter lengths and lower computational rates compared to standard single-rate filter designs, thereby providing a practical solution

to an otherwise difficult problem. These advantages are usually at the expense of increased filter group delay.

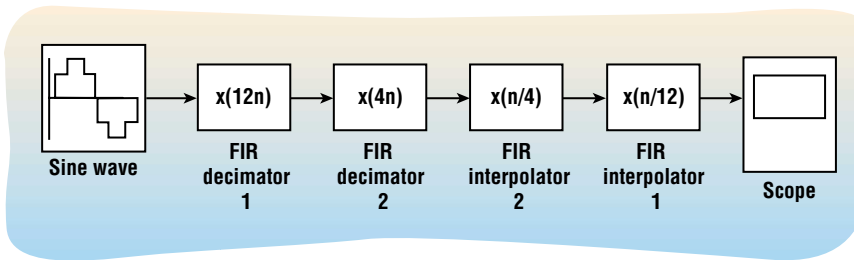
Multirate FIR filters can leverage many standard FIR filter-design methods. To illustrate the design process, there are a number of examples of multirate filters for lowpass, highpass, and bandpass filters with very narrow passbands by using MATLAB and the Signal Processing Toolbox for filter design, as well as Simulink and the DSP Blockset for implementation. The concepts for designing narrow filters can be extended to the design of filters with very-wide passbands.



1. Lowpass FIR filter design specifications can be seen in this figure.

Consider the design of a lowpass filter that preserves frequencies below 70 Hz, removes frequencies above 80 Hz, and runs at an input sample rate of 8 kHz. In terms of normalized frequencies, the filter cutoff is 80/4000, or 0.02, where normalizing is accomplished with respect to half the sample rate. To completely specify a FIR filter, one needs to specify the maximum

passband ripple (for this example, 0.15 dB or 0.01 linear) and minimal stopband attenuation (80 dB or 10^{-4} linear)



2. This Simulink model implements a lowpass, narrow passband filter as a two-stage multirate FIR filter.

that should be achieved (Fig. 1).

For a single-rate equiripple FIR filter, the Signal Processing Toolbox function `remezord` is used to estimate the filter order:

```
>> remezord[(70 80), (1 0), (0.01 1E-4), 8000]
```

The filter design requires an order 2511 (2512-coefficient) FIR filter. To design the actual filter, proceed as follows:

```
>> b = remez[2511, (0 70 80 4000)/4000, (1 1 0 0), (1 50)];
```

Assuming one multiply-and-accumulate (MAC) operation is required for each pair of symmetric filter coefficients, the required computational rate is $2512/2 \text{ MAC/sample} \times 8000 \text{ samples/s}$ or slightly more than 10 million MAC/s. This computational rate serves as a reference to the benefits of multirate filtering.

The bandwidth of the signal output from the single-rate filter is very narrow compared to the sample rate. The minimum sample rate necessary to reconstruct a signal with 80-Hz bandwidth is 160 Hz, which requires only $160/8000$, or $1/50$ the present sample rate.

Pursuing decimation further, cascading an FIR filter with a decimator is most efficiently performed with a polyphase FIR decimation filter. This reduces the computational rate by the decimation factor M . To understand the rate reduction, consider that only one out of every M input samples is outputted by the decimator. All others are “thrown away.” Thus, it is possible to consider computing one out of every M output samples. Polyphase FIR decimation filters are structured to balance the computation of this one output sample over M consecutive input samples to achieve the noted reduction in computational rate.

FILTER EFFICIENCY

However, decimation represents only one part of the achievable improvement in filter efficiency. For general multirate filtering, the output rate must be equal to the input rate. Thus, it is necessary to interpolate the output of the polyphase FIR decimation filter back up to the original input rate. This is accomplished by cascading an up-

sample operation (which inserts $L-1$ zeros after every sample) followed by an FIR anti-imaging filter. This anti-imaging filter is most efficiently implemented using a polyphase FIR interpolation filter. Similar to the decimator, this implementation accomplishes the interpolation while reducing the computational rate by the interpolation factor

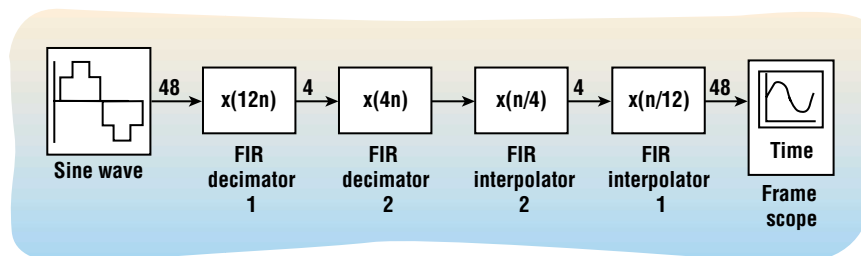
L . This outcome is best understood by considering that one out of every L interpolated samples is non-zero prior to entering the interpolation filter. Filter computation for the zero-valued samples is not necessary, further reducing the computation rate.

Single-stage multirate filters: Based on this development, it is clear that the cascaded polyphase FIR decimation and interpolation filters form efficient, multirate filters. While a single-stage multirate filter does nothing to ease the requirement on filter length it can go a long way toward reducing the computational load. Returning to the lowpass filter example, it is possible to choose to decimate the filter output by a factor of 48 (slightly smaller than the maximum decimation factor of $8000/160$, or 50, to provide a margin of safety).

Multistage multirate filters: Additional efficiency can be achieved by cascading several multirate stages to reduce the total number of filter coefficients. For example, instead of implementing a design based on a single decimation factor of 48, consider implementing two stages of decimation using 12 and 4, or 24 and 2, or any combination of integer factors that multiply to 48. If the factors 12 and 4 are chosen, the resulting Simulink model is shown in Fig. 2.

THE FIRST STAGE

The first stage decimates by a factor of 12. Thus, the corresponding first-stage filter must cutoff before $1/12$ of the normalized sampling frequency (approximately 0.083) to prevent aliasing. This cutoff requirement is less constraining than the overall specification that requires an 80-Hz cutoff. (Recall that an 80-Hz cutoff translates to $80/4000$, or 0.02, in normalized frequency, which is smaller and, therefore, more constraining than 0.083.) This leads to a lower-order first-stage filter design. The second-stage filter is then responsible for attaining our final filter requirements (70-Hz passband, 80-Hz stopband), while operating at $1/12$ the original sample rate. As a result, it is



3. The sine-wave block produces 48-element frames of data. Sample rate changes are revealed by corresponding changes in frame size.

possible to ease the design constraints placed on the second stage, resulting in a lower-order second-stage filter as well. However, since two filters are now being cascaded, the passband ripple requirement for each filter stage is half that of the single-stage filter. Reducing the ripple specification for each stage tends to increase overall filter length. The choice of decimation factor for each stage supports a degree of optimization across these design factors, and can lead to a design that is more efficient than single-stage designs.

Using the equiripple filter design technique that was used in the single-stage filter examples, the estimated first stage filter order is 225. It was computed using:

$$\gg \text{remezord} [(1/12-0.03 \ 1/12), (1 \ 0), (0.005 \ 1e-4), 2]$$

The passband ripple specification has been reduced from 0.01 to 0.005, and was chosen with the width of the transition band (0.03) to avoid encroaching on the desired final passband (80-Hz bandwidth). The second-stage filter requires (coincidentally) another order 225 filter:

$$\gg \text{remezord} [(70 \ 80), (1 \ 0), (0.5e-2 \ 1e-4), 8000/12]$$

Using cascaded multirate stages in this design therefore requires 452 coefficients, which are significantly fewer than the 2512 coefficients in the single-stage design. The computation rate for the decimator stage of the two-stage design is:

$$\begin{aligned} & (226/2 \text{ MAC/sample}) \times (8000 \text{ sample/s}) \times (1/12 \text{ decimation factor}) \\ & + (226/2 \text{ MAC/sample}) \times (8000/12 \text{ sample/s}) \times (1/4 \text{ decimation factor}). \end{aligned}$$

The interpolator is again identical to the decimator, leading to a total of 188 kMAC/s.

A comparison of the single-rate, single-stage multirate, and two-stage multirate design examples appear in the **table**. The two-stage multirate filter design can be further improved to reduce the filter order and computational rate by optimizing the choice of decimation factors.

Finally, frame-based implementation methods are available to further increase the throughput of a multirate

Comparing the design examples

| Design techniques | Number of FIR coefficients | Computational 1 rate (kMAC/s) | Group delay (samples) |
|------------------------|----------------------------|-------------------------------|-----------------------|
| Single rate | 2512 | 10,048 | 1256 |
| Single stage multirate | 2512 | 419 | 1256 |
| Two stage multirate | 452 | 188 | 2925 |

model. The sine-wave block in **Fig. 3** produces 48-element frames of data; sample rate changes are revealed by corresponding changes in frame size. Thus, decimation by 12 in the first filter stage yields an output frame size of 48/12 or 4 samples. The size of the frame input to the filter must be a multiple of the aggregate decimation factor (here, it is 48). All of the multirate techniques explored in this article may be implemented using sample-based or frame-based methods.

NARROWBAND FILTERS

The techniques used for designing narrow passband lowpass filters may be readily extended to bandpass and highpass filter designs. Multirate techniques can be leveraged to achieve a bandpass filter with a narrow passband by employing quadrature modulation. In quadrature modulation, the center frequency of the passband is modulated to baseband (zero frequency), filtered by a narrow lowpass filter, then remodulated to the original center frequency. Quadrature modulation requires that the input signal entering the lowpass filter contain real and imaginary parts. This increases the computational load because both parts of the signal need to be processed.

The modulation and demodulation operations are realized by multiplying by a discrete complex exponential modulation function, $m(n)$, which is sampled at F_s sample/s and generated at f , the center frequency of interest:

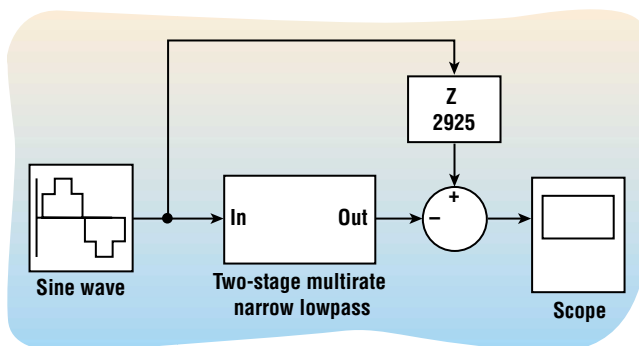
$$m(n) = \cos\left(\frac{2\pi fn}{F_s}\right) + j \sin\left(\frac{2\pi fn}{F_s}\right) \quad (1)$$

A highpass filter with narrow passband is an extension of the bandpass design, with the center frequency f set to half the sample rate ($F_s/2$). In effect, the narrow lowpass filter is modulated to half the sample rate and becomes a narrow highpass filter. In this case, the modulation function degenerates to:

$$\begin{aligned} m(n) &= \cos(\pi n) + \\ & j \sin(\pi n) = \cos(\pi n) \quad (2) \end{aligned}$$

The modulation function is no longer complex, and the values of the cosine terms are simply $(-1)^n$. Moreover, instead of multiplying the input signal by $m(n)$, the designer can choose to multiply the corresponding FIR filter coefficients by the same alternating sequence of +1 and -1, effectively modulating the filter itself to the frequency band of interest. Thus, implementing a multirate narrow highpass filter is identical to implementing a narrow lowpass filter, with the sign of every other coefficient negated.

The multirate, narrow passband filter designs can be used to develop highpass and lowpass filters with



4. This diagram depicts the multistage, multirate FIR filter implementation of a highpass filter with wide passband.

wide passbands. Assuming the transfer function of a low-pass filter is $L(z)$, the transfer function of a corresponding highpass filter with a wide passband is $H(z) = 1 - L(z)$. Essentially, the lowpass filter response is subtracted from an allpass filter response. The result is a wide highpass filter.

Subtracting the multirate lowpass filter output from a delayed replica of the original input signal (**Fig. 4**) results in a wide highpass filter. Take care to select the delay to exactly equal the group delay of the multirate filter. For a single-rate FIR filter, the group delay is half the filter length. For a two-stage, multirate, lowpass design, the group delay of each filter must be calculated relative to the original input rate by multiplying the single-rate group delay values by the preceding decimation factors. Recalling that there is a decimation and interpolation filter stage, the

final group delay N of the earlier two-stage lowpass filter design is:

In a similar manner, a wideband lowpass filter can be

$$N = 225 + 225 \times 12 = 2925 \quad (3)$$

designed from a narrow highpass filter. The design and implementation of efficient filters with very-narrow or very-wide passbands can be achieved using multistage, multirate DSP filtering techniques. *WSD*

DON OROFINO, PAUL PACHECO, and NICK ADLER, *DSP Development, The MathWorks, Inc., 3 Apple Hill Drive, Natick, MA 01760; (508) 647-7000, FAX: (508) 647-7001, WWW: <http://www.mathworks.com>.*