Understanding cascaded integrator-comb filters

Richard Lyons

MARCH 31, 2005

The previously obscure CIC filter is now vital to many high-volume wireless communications tasks and equipment. Using CIC filters can cut costs, improve reliability, and help performance. Here's a primer to get you started.

Cascaded integrator-comb (CIC) digital filters are computationally efficient implementations of narrowband lowpass filters and are often embedded in hardware implementations of decimation and interpolation in modern communications systems. CIC filters were introduced to the signal-processing community, by Eugene Hogenauer, more than two decades ago, but their application possibilities have grown in recent years.[1] Improvements in chip technology, the increased use of polyphase filtering techniques, advances in delta-sigma converter implementations, and the significant growth in wireless communications have all spurred much interest in CIC filters.

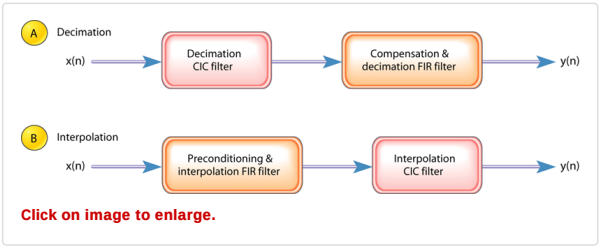
While the behavior and implementation of these filters isn't complicated, their coverage has been scarce in the literature of embedded systems. This article attempts to augment the body of literature for embedded systems engineers. After describing a few applications for CIC filters, I'll introduce their structure and behavior, present the frequency-domain performance of CIC filters, and discuss several important practical issues in building these filters.

CIC filter applications

CIC filters are well-suited for antialiasing filtering prior to decimation (sample-rate reduction), as shown in Figure 1a and for anti-imaging filtering for interpolated signals

(sample-rate increase) as in Figure 1b. Both applications are associated with very high-data-rate filtering, such as hardware quadrature modulation and demodulation in modern wireless systems and delta-sigma A/D and D/A converters.

Figure 1: CIC filter applications



Because their frequency-magnitude-response envelopes are sin(x)/x-like, CIC filters are typically either followed or preceded by higher performance linear-phase lowpass tapped-delay-line FIR filters whose tasks are to compensate for the CIC filter's non-flat passband. That cascaded-filter architecture has valuable benefits. For example, with decimation, you can greatly reduce computational complexity of narrowband lowpass filtering compared with if you'd used a single lowpass finite impulse response (FIR) filter. In addition, the follow-on FIR filter operates at reduced clock rates minimizing power consumption in high-speed hardware applications.

A crucial bonus in using CIC filters, and a characteristic that makes them popular in hardware devices, is that they require no multiplication. The arithmetic needed to implement these digital filters is strictly additions and subtractions only. With that said, let's see how CIC filters operate.

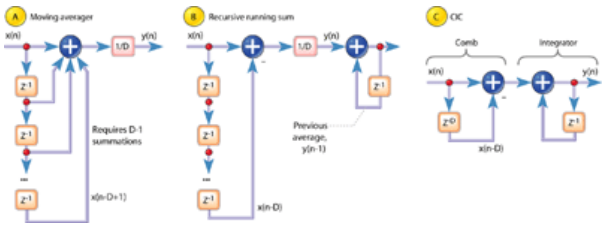


Figure 2: D-point averaging filters
View full-sized image

Recursive running-sum filter

CIC filters originate from the notion of a *recursive running-sum filter*, which is itself an efficient form of a nonrecursive *moving averager*. Recall the standard D-point moving-average process in Figure 2a. There we see that D-1 summations (plus one multiply by 1/D) are necessary to compute the averager output $y(n)$.

The D-point moving-average filter's output in time is expressed as:

$$y(n) = \frac{1}{D}\left[\begin{array}{l} x(n)+x(n-1) \\ +x(n-2) \\ +x(n-3)+\ldots \\ +x(n-D+1) \end{array}\right]$$

Equation 1

where n is our time-domain index. The z-domain expression for this moving averager is:

$$Y(z) = \frac{1}{D}\left[\begin{array}{l} X(z)+X(z)z^{-1} \\ +X(z)z^{-2}\ldots \\ +X(z)z^{-D+1} \end{array}\right]$$

Equation 2

while its z-domain $H(z)$ transfer function is:

$$H(z) = \frac{Y(z)}{X(z)}$$
$$= \frac{1}{D}\left[1 + z^{-1} + z^{-2} + \ldots + z^{-D+1}\right]$$
$$= \frac{1}{D}\sum_{n=0}^{D-1} z^{-n}.$$

Equation 3

I provide these equations not to make things complicated, but because they're useful. Equation 1 tells us how to build a moving averager, and Equation 3 is in the form used by commercial signal-processing software to model the frequency-domain behavior of the moving averager.

The next step in our journey toward understanding CIC filters is to consider an equivalent form of the moving averager, the *recursive running-sum filter* depicted in Figure 2b. There we see that the current input sample $x(n)$ is added, and the oldest input sample $x(n\text{-}D)$ is subtracted from the previous output average $y(n\text{-}1)$. It's called "recursive" because it has feedback. Each filter output sample is retained and used to compute the next output value. The recursive running-sum filter's difference equation is:

$$y(n) = \frac{1}{D}\left[x(n) - x(n-D)\right] + y(n-1)$$

Equation 4

having a z-domain $H(z)$ transfer function of:

$$H(z) = \frac{1}{D}\frac{1 - z^{-D}}{1 - z^{-1}}.$$

Equation 5

We use the same $H(z)$ variable for the transfer functions of the moving-average filter and the recursive running-sum filter because their transfer functions are equal to each other! It's true. Equation 3 is the nonrecursive expression and Equation 5 is the recursive expression for a D-point averager. The mathematical proof of this can be found in my book on digital signal processing, but shortly I'll demonstrate that equivalency with an example.[2]

Here's why we care about recursive running-sum filters: the standard moving averager in Figure 2a must perform D-1 additions per output sample. The recursive running-sum filter has the sweet advantage that only one addition and one subtraction are required per output sample, regardless of the delay length D. This computational efficiency makes the recursive running-sum filter attractive in many applications seeking noise reduction through averaging. Next we'll see how a CIC filter is, itself, a recursive running-sum filter.

CIC filter structures

If we condense the delay-line representation and ignore the 1/D scaling in Figure 2b we obtain the classic form of a 1st-order CIC filter, whose cascade structure is shown in Figure 2c. The feedforward portion of the CIC filter is called the *comb* section, whose differential delay is D, while the feedback section is typically called an *integrator*. The comb stage subtracts a delayed input sample from the current input sample, and the integrator is simply an accumulator. The CIC filter's difference equation is:

$$y(n) = x(n) - x(n-D) + y(n-1)$$

Equation 6

and its z-domain transfer function is:

$$H_{cic}(z) = \frac{1 - z^{-D}}{1 - z^{-1}}.$$

Equation 7

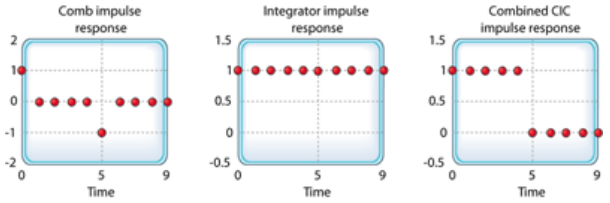


Figure 3: Single-stage CIC filter time-domain responses when D = 5
View full-sized image

To see why the CIC filter is of interest, first we examine its time-domain behavior, for D = 5, shown in Figure 3. If a unit-impulse-sequence, a unity-valued sample followed by many zero-valued samples, was applied to the comb stage, that stage's output is as shown in Figure 3a. Now think, what would be the output of the integrator if its input was the comb stage's impulse response? The initial positive impulse from the comb filter starts the integrator's all-ones output, as in Figure 3b. Then, D samples later, the negative impulse from the comb stage arrives at the integrator to zero all further CIC filter output samples.

The key issue is that the combined unit-impulse response of the CIC filter, being a rectangular sequence, is identical to the unit-impulse responses of a moving-average filter and the recursive running-sum filter. (Moving averagers, recursive running-sum filters, and CIC filters are close kin. They have the same z-domain pole/zero locations, their frequency magnitude responses have identical shapes, their phase responses are identical, and their transfer functions differ only by a constant scale factor.) If you understand the time-domain behavior of a moving averager, then you now understand the time-domain behavior of the CIC filter in Figure 2c.

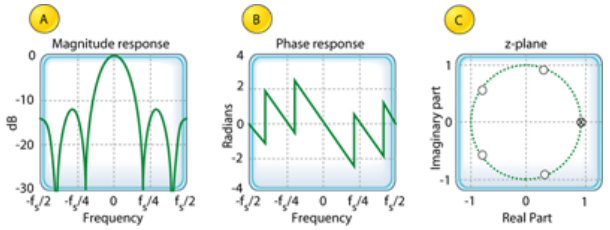


Figure 4: Characteristics of a single-stage CIC filter when D = 5
View full-sized image

The frequency magnitude and linear-phase response of a D = 5 CIC filter is shown in Figure 4a where the frequency f_s is the input-signal sample rate in Hz.

We can obtain an expression for the CIC filter's frequency response by evaluating Equation 7's $H_{cic}(z)$ transfer function on the z-plane's unit circle, by setting $z = e^{j2\pi f}$, yielding:

$$H_{cic}\left(e^{j2\pi f}\right) = \frac{1 - e^{-j2\pi fD}}{1 - e^{-j2\pi f}} = \frac{e^{-j2\pi fD/2}\left(e^{j2\pi fD/2} - e^{-j2\pi fD/2}\right)}{e^{-j2\pi f/2}\left(e^{j2\pi f/2} - e^{-j2\pi f/2}\right)}$$

Equation 8

Using Euler's identity $2j\sin(\alpha) = e^{j\alpha} - e^{j\alpha}$, we can write:

$$H_{cic}\left(e^{j2\pi f}\right) = \frac{e^{-j2\pi fD/2}}{e^{-j2\pi f/2}} \frac{2j\sin(2\pi fD/2)}{2j\sin(2\pi f/2)}$$

$$= e^{-j2\pi f(D-1)/2} \frac{\sin(\pi fD)}{\sin(\pi f)}.$$

Equation 9

If we ignore the phase factor in Equation 9, that ratio of sin() terms can be approximated by a $\sin(x)/x$ function. This means the CIC filter's frequency magnitude response is approximately equal to a $\sin(x)/x$ function centered at 0Hz as we see in Figure 4a. (This is why CIC filters are sometimes called *sinc* filters.)

Digital-filter designers like to see z-plane pole/zero plots, so we provide the z-plane characteristics of a D = 5 CIC filter in Figure 4c, where the comb filter produces D zeros, equally spaced around the unit-circle, and the integrator produces a single pole canceling the zero at z = 1. Each of the comb's zeros, being a Dth root of 1, are located at $z(m)$ = j

$e^{2\pi ml/D}$, where m = 0, 1, 2, ..., D-1, corresponding to a magnitude null in Figure 4a.

The normally risky situation of having a filter pole directly on the unit circle need not trouble us here because there is no coefficient quantization error in our $H_{cic}(z)$ transfer function. CIC filter coefficients are ones and can be represented with perfect precision with fixed-point number formats. Although recursive, happily CIC filters are guaranteed stable, linear-phase shown in Figure 4b, and have finite-length impulse responses. At 0Hz (DC) the gain of a CIC filter is equal to the comb filter delay D. This fact, whose derivation is available, will be important to us when we actually implement a CIC filter in hardware.[2]

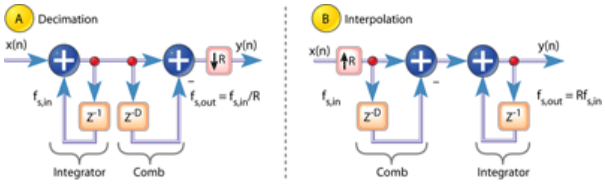


Figure 5: Single-stage CIC filters used in decimation and interpolation
View full-sized image

Again, CIC filters are primarily used for antialiasing filtering prior to decimation and for anti-imaging filtering for interpolated signals. With those notions in mind we swap the order of Figure 2c's comb and integrator—we're permitted to do so because those operations are linear—and include decimation by a sample rate change factor R in Figure 5a. (You may wish to prove that the unit-impulse response of the integrator/comb combination, prior to the sample rate change, in Figure 5a is equal to that in Figure 3c.) In most CIC filter applications the rate change R is equal to the comb's differential delay D, but we'll keep them as separate design parameters for now.

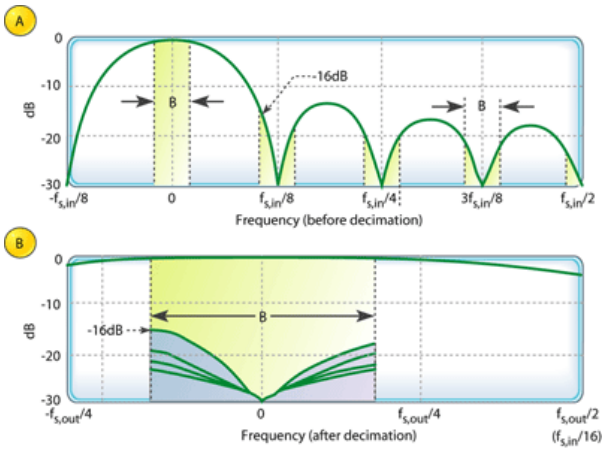


Figure 6: Magnitude response of a 1st-order, D = 8, decimating CIC filter: before decimation; aliasiing after R = 8 decimation
View full-sized image

The decimation operation $\downarrow R$ means discard all but every Rth sample, resulting in an output sample rate of f_s,out = $f_{s,in}/R$. To investigate a CIC filter's frequency-domain behavior in more detail, Figure 6a shows the frequency magnitude response of a D = 8 CIC filter prior to decimation. The spectral band, of width B, centered at 0Hz is the desired passband of the filter. A key aspect of CIC filters is the spectral folding that takes place due to decimation.

Those B-width shaded spectral bands centered about multiples of $f_{s,in}/R$ in Figure 6a will alias directly into our desired passband after decimation by R = 8 as shown in Figure 6b. Notice how the largest aliased spectral component, in this example, is roughly 16dB below the peak of the band of interest. Of course the aliased power levels depend on the bandwidth B—the smaller B is, the lower the aliased energy after decimation.

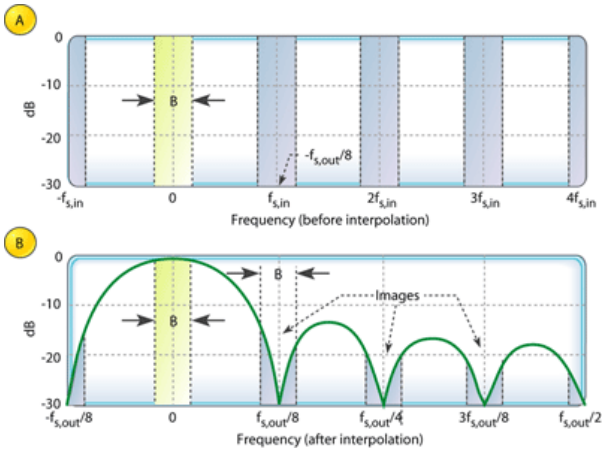


Figure 7: 1st-order, $D = R = 8$, interpolating CIC filter spectra: input spectrum; output spectral images
View full-sized image

Figure 5b shows a CIC filter used for interpolation where the $\uparrow R$ symbol means insert R-1 zeros between each $x(n)$ sample, yielding a $y(n)$ output sample rate of f_S,out = Rf_S,in. (In this CIC filter discussion, interpolation is defined as zeros-insertion followed by filtering.) Figure 7a shows an arbitrary baseband spectrum, with its spectral replications, of a signal applied to the $D = R = 8$ interpolating CIC filter of Figure 5b. The filter's output spectrum in Figure 7b shows how imperfect filtering gives rise to the undesired spectral images.

After interpolation, unwanted images of the B-width baseband spectrum reside at the null centers, located at integer multiples of $f_{s,\text{out}}/R$. If we follow the CIC filter with a traditional lowpass tapped—delay-line FIR filter, whose stopband includes the first image band, fairly high image rejection can be achieved.

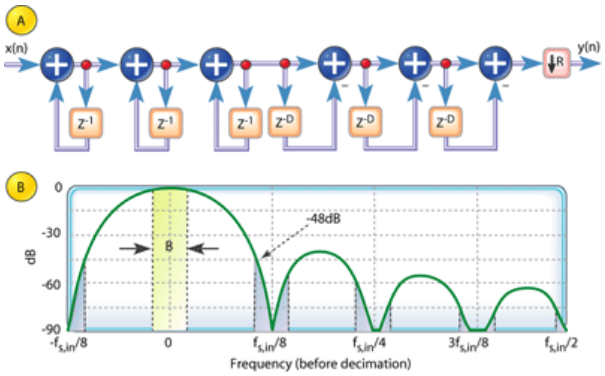


Figure 8: 3rd-order CIC decimation filter structure, and magnitude response before decimation when $D = R = 8$
View full-sized image

Improving CIC attenuation

The most common method to improve CIC filter anti-aliasing and image-reject attenuation is by increasing the order M of the CIC filter using multiple stages. Figure 8 shows the structure and frequency magnitude response of a 3rd-order ($M = 3$) CIC decimating filter.

Notice the increased attenuation at $f_{s,\text{out}}/R$ in Figure 8b compared with the 1st-order CIC filter in Figure 6a. Because the $M = 3$ CIC stages are in cascade, the overall frequency magnitude response will be the product of their individual responses or:

$$\left|H_{\text{cic,Mth-order}}\left(e^{j2\pi f}\right)\right| = \left|\frac{\sin(\pi fD)}{\sin(\pi f)}\right|^M.$$

Equation 10

The price we pay for improved anti-alias attenuation is additional hardware adders and increased CIC filter passband droop. An additional penalty of increased filter order comes from the gain of the filter, which is exponential with the order. Because CIC filters generally must work with full precision to remain stable, the number of bits in the adders is $M\log_2(D)$, which means a large data word-width penalty for higher order filters. Even

so, this multistage implementation is common in commercial integrated circuits, where an Mth-order CIC filter is often called a sinc^M filter.

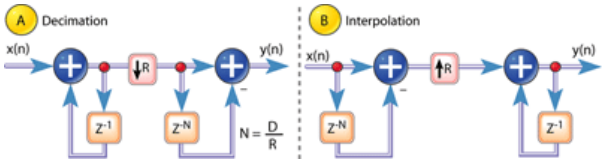


Figure 9: Single-stage CIC filter implementations: for decimation; for interpolation
View full-sized image

Building a CIC filter

In CIC filters, the comb section can precede, or follow, the integrator section. It's sensible, however, to put the comb section on the side of the filter operating at the lower sample rate to reduce the storage requirements in the delay. Swapping the comb filters from Figure 5 with the rate-change operations results in the most common implementation of CIC filters, as shown in Figure 9. Notice the decimation filter's comb section now has a delay length (differential delay) of $N = D/R$. That's because an N-sample delay after decimation by R is equivalent to a D-sample delay before decimation by R. Likewise for the interpolation filter; an N-sample delay before interpolation by R is equivalent to a D-sample delay after interpolation by R.

Those Figure 9 configurations yield two major benefits: first, the comb section's new differential delay is decreased to $N = D/R$ reducing data storage requirements; second, the comb section now operates at a reduced clock rate. Both of these effects reduce hardware power consumption.

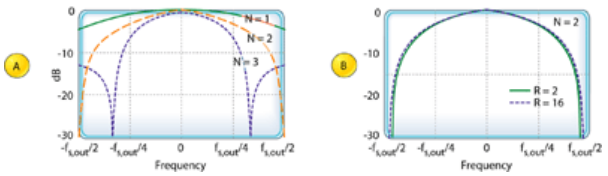


Figure 10: CIC decimation filter responses: for various values of differential delay N, when R = 8; for two decimation factors when N =2
View full-sized image

The comb section's differential delay design parameter N is typically 1 or 2 for high sample-rate ratios as is often used in up/down-converters. N effectively sets the number of nulls in the frequency response of a decimation filter, as shown in Figure 10a.

An important characteristic of a CIC decimator is that the shape of the filter response changes very little, as shown in Figure 10b, as a function of the decimation ratio. For values of R larger than roughly 16, the change in the filter shape is negligible. This allows the same compensation FIR filter to be used for variable-decimation ratio systems.

The CIC filter suffers from register overflow because of the unity feedback at each integrator stage. The overflow is of no consequence as long as the following two conditions are met:

- the range of the number system is greater than or equal to the maximum value expected at the output, and
- the filter is implemented with two's complement (nonsaturating) arithmetic.

Because a 1st-order CIC filter has a gain of $D = NR$ at 0Hz (DC), M cascaded CIC decimation filters have a net gain of $(NR)^M$. Each additional integrator must add another NR bits width for stages. Interpolating CIC filters have zeros inserted between input samples reducing its gain by a factor of $1/R$ to account for the zero-valued samples, so the net gain of an interpolating CIC filter is $(NR)^M/R$. Because the filter must use integer arithmetic, the word widths at each stage in the filter must be wide enough to accommodate the maximum signal (full-scale input times the gain) at that stage.

Although the gain of an Mth-order CIC decimation filter is $(NR)^M$ individual integrators can experience overflow. (Their gain is infinite at DC.) As such, the use of two's complement arithmetic resolves this overflow situation just so long as the integrator word width accommodates the maximum difference between any two successive samples (in other words, the difference causes no more than a single overflow). Using the two's complement binary format, with its modular wrap-around property, the follow-on comb filter will properly compute the correct difference between two successive integrator output samples.

For interpolation, the growth in word size is one bit per comb filter stage and overflow must be avoided for the integrators to accumulate properly. So, we must accommodate

an extra bit of data word growth in each comb stage for interpolation. There is some small flexibility in discarding some of the least significant bits (LSBs) within the stages of a CIC filter, at the expense of added noise at the filter's output. The specific effects of this LSB removal are, however, a complicated issue; you can learn more about the issue by reading Hogenauer's paper.[1]

While the preceding discussion focused on hard-wired CIC filters, these filters can also be implemented with programmable fixed-point DSP chips. Although those chips have inflexible data paths and word widths, CIC filtering can be advantageous for high sample-rate changes. Large word widths can be accommodated with multiword additions at the expense of extra instructions. Even so, for large sample-rate change factors the computational workload per output sample, in fixed-point DSP chips, may be small.

Compensation filters

In typical decimation/interpolation filtering applications we want reasonably flat passband and narrow transition-region filter performance. These desirable properties are not provided by CIC filters alone, with their drooping passband gains and wide transition regions. We alleviate this problem, in decimation for example, by following the CIC filter with a compensation nonrecursive FIR filter, as in Figure 1a, to narrow the output bandwidth and flatten the passband gain.

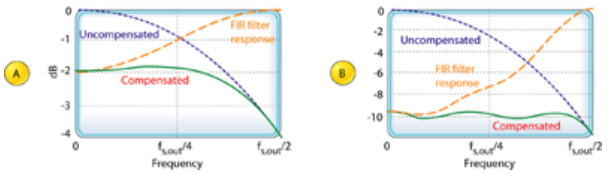


Figure 11: Compensation FIR filter responses; with a 1st-order decimation CIC filter; with a 3rd-order decimation
View full-sized image

The compensation FIR filter's frequency magnitude response is ideally an inverted version of the CIC filter passband response similar to that shown by the dashed curve in Figure 11a for a simple three-tap FIR filter whose coefficients are [-1/16, 9/8, -1/16]. With the dotted curve representing the uncompensated passband droop of a 1st-order R = 8 CIC filter, the solid curve represents the compensated response of the cascaded filters. If either the passband bandwidth or CIC filter order increases the correction becomes greater, requiring more compensation FIR filter taps. An example of this situation is shown in Figure 11b where the dotted curve represents the passband droop of a 3rd-order R = 8 CIC filter and the dashed curve, taking the form of [x/sin(x)]3, is the response of a 15-tap compensation FIR filter having the coefficients [-1, 4, -16, 32, -64, 136, -352, 1312, -352, 136, -64, 32, -16, 4, -1].

A wideband correction also means signals near $f_{s,out}/2$ are attenuated with the CIC filter and then must be amplified in the correction filter, adding noise. As such, practitioners often limit the passband width of the compensation FIR filter to roughly 1/4 the frequency of the first null in the CIC-filter response.

Those dashed curves in Figure 11 represent the frequency magnitude responses of compensating FIR filters within which no sample-rate change takes place. (The FIR filters' input and output sample rates are equal to the $f_{s,out}$ output rate of the decimating CIC filter.) If a compensating FIR filter were designed to provide an additional decimation by two, its frequency magnitude response would look similar to that in Figure 12, where <>$_{s,in}$ is the compensation filter's input sample rate.

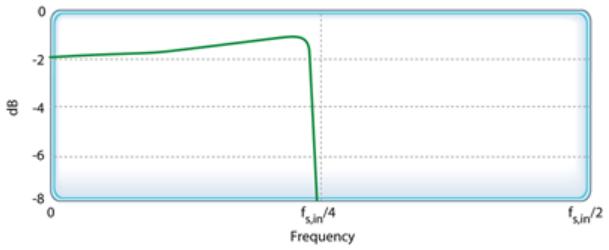


Figure 12: Frequency magnitude response of a decimate-by-2 compensation FIR filter

View full-sized image

Advanced techniques

Here's the bottom line of our CIC-filter discussion: a decimating CIC filter is merely a very efficient recursive implementation of a moving-average filter, with NR taps, whose output is decimated by R. Likewise, the interpolating CIC filter is insertion of R-1 zero samples between each input sample followed by an NR-tap moving-average filter running at the output sample rate $f_{s,out}$. The cascade implementations in Figure 1 result in total computational workloads far less than using a single FIR filter alone for high sample-rate-change decimation and interpolation. CIC filter structures are designed to maximize the amount of low-sample-rate processing to minimize power consumption in high-speed hardware applications. Again, CIC filters require no multiplication; their arithmetic is strictly addition and subtraction. Their performance allows us to state that, technically speaking, CIC filters are lean, mean filtering machines.

In closing, there are ways to build nonrecursive CIC filters that ease the word-width growth problem of the traditional recursive CIC filters. Those advanced CIC filter architectures are discussed in my book *Understanding Digital Signal Processing, 2E*.[2]

Richard Lyons is a consulting systems engineer and lecturer with Besser Associates in Mountain View, Ca. He is the author of *Understanding Digital Signal Processing 2/E* and an associate editor for the *IEEE Signal Processing Magazine* where he created and edits the "DSP Tips & Tricks" column. You can reach him at r.lyons@ieee.org.

Endnotes

1. Hogenauer, Eugene. "An Economical Class of Digital Filters For Decimation and Interpolation," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-29, pp. 155-162, April 1981.
2. Lyons, Richard, *Understanding Digital Signal Processing, 2nd Ed.*, Prentice Hall, Upper Saddle River, New Jersey, 2004, pp. 556-561.