Randy Yates
and Richard Lyons

DC Blocker Algorithms

"DSP Tips and Tricks" introduces practical design and implementation signal processing algorithms that you may wish to incorporate into your designs. We welcome readers to submit their contributions to the Associate Editors Rick Lyons (r.lyons@ieee.org) or Britt Rorabaugh (dspboss@aol.com).

The removal of a dc bias (a constant-amplitude component) from a signal is a common requirement in signal processing systems. Thus, a good dc blocking algorithm is a desirable tool to have in one's bag of signal processing tricks. In this article we present both a nonlinear phase fixed-point dc blocker (using a noise-shaping trick that eliminates a signal's dc bias using fixed-point arithmetic) and a general linear-phase dc blocker network that may prove useful in various DSP applications.

FIXED-POINT DC BLOCKER

Simple fixed-point implementations of a dc blocker algorithm have a vexing quantization problem that can create more dc bias than they block. In this section we present a "leaky integrator" and a noise-shaping trick called "fraction-saving" to eliminate the quantization problem when using fixed-point arithmetic.

The simplest filter that blocks dc is the digital differentiator, whose transfer function is given by

$$R(z) = 1 - z^{-1}. \tag{1}$$

Having a z-plane zero at $z = 1$, the differentiator has infinite attenuation at 0 Hz and perfectly blocks dc. However, the differentiator also attenuates spectral components close to dc as shown by the dashed curve in Figure 1.

The standard method of shoring up that drooping frequency response of a differentiator is to place a pole just inside the z-plane zero at $z = 1$ using a single-pole filter whose transfer function is

$$S(z) = \frac{1}{1 - pz^{-1}}, \tag{2}$$

where p is a real pole and $0 < p < 1$. Equation (2) describes a leaky integrator, a nonideal integrator that leaks some energy away rather than perfectly integrating dc. The integrator's response to a dc input will not be an ever-increasing output, but rather an output that increases for a time and then levels off. The term "leaky" is carried over from analog design in which the capacitor used to implement an integrator was imperfect due to the flow of leakage current. The cascaded differentiator/integrator transfer function is given by

$$H(z) = R(z)S(z) = \frac{1 - z^{-1}}{1 - pz^{-1}}. \tag{3}$$

The location of the pole $z = p$ of this nonlinear-phase filter presents a tradeoff between the filter's bandwidth and time-domain transient response. The magnitude responses of $H(z)$ for various values of p are shown in Figure 1.

Cascaded filters implementing (3) work fine in a floating-point number system and have been described in the literature [1]–[2]. Next we discuss a potential problem in fixed-point implementations of (3).

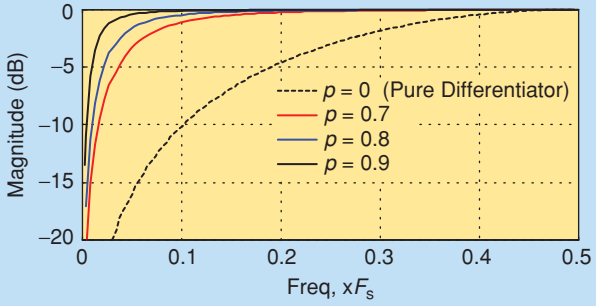
FIXED-POINT IMPLEMENTATION

To understand the effects of fixed-point arithmetic on the dc blocker algorithm in (3), let's consider the differentiator and leaky integrator sections independently using the network in Figure 2(a). Assume we are using 16-b input and output binary words as indicated in the figure.

The differentiator's difference equation is

$$w[n] = x[n] - x[n-1]. \tag{4}$$

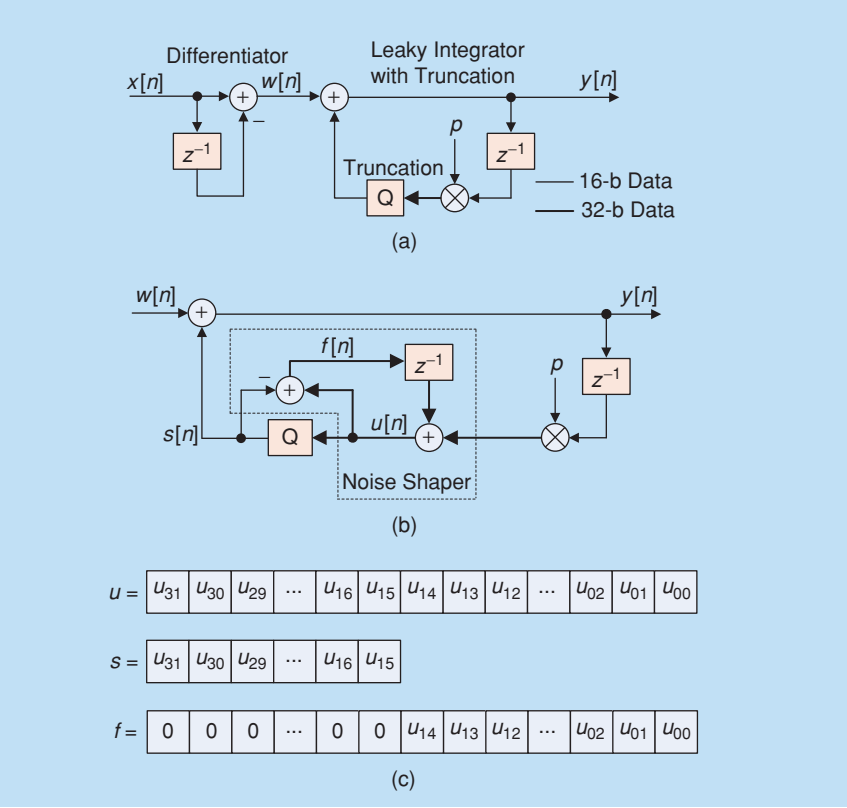
Note that, at most, the computation in (4) will require one more bit than the input (17 b total) to avoid possible overflow, which happens when the input



[FIG1] Cascaded differentiator/integrator frequency response.

Digital Object Identifier 10.1109/MSP.2007.914713

1053-5888/08/$25.00©2008IEEE



[FIG2] **Fixed-point dc blocking filter: (a) simple structure, (b) optimized integrator structure, and (c) bit-wise representation of u, s, and f.**

changes sign and the magnitude of the difference is greater than 32,768. However, barring that case, we can implement the differentiator without requiring extra precision in the intermediate computation and thus avoid the associated requantization to 16 b at the output. Also note that the differentiator is nonrecursive, so quantization effects, if they do occur, are not circulated back through the differentiator.

The leaky integrator has a difference equation of

$$y[n] = p \cdot y[n-1] + w[n]. \qquad (5)$$

Because the goal is to implement (5) on a fixed-point (integer) processor, its terms must each be represented as an integer and the operations must be performed using integer arithmetic. If we want the $x[n]$ input and $y[n]$ output to be scaled identically, then we must quantize (truncate) the 32-b product result in (5) to 16 b. It is precisely this quantization that introduces a potentially significant dc offset error into the $y[n]$ output, and the closer the pole is to the unit circle the larger the potential dc error.

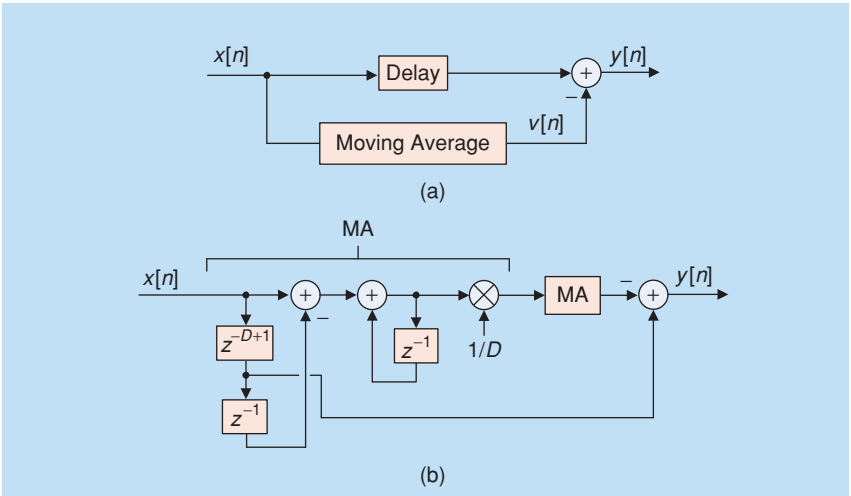
Like any quantizer, the quantization's dc error can be shaped by placing feedback around the quantizer. In order to implement our noise-shaping trick, all that is required is to save the fraction of the quantizer input and feed it back in the next update. Hence the term "fraction-saving." This process is shown in Figure 2(b). Note that sequence $f[n] = u[n]–s[n]$ is the fractional part of $u[n]$ in two's complement arithmetic after quantization by truncation. This can be illustrated as follows: let u be represented bit-wise as shown in Figure 2(c). Notice that s is u with the 15 least significant bits truncated. So $f = u–s$ is the fractional part of u.

Using the fraction-saving scheme in Figure 2(b) guarantees that the 16-b $y[n]$ output will have a dc bias of zero [4]. (Detailed descriptions of noise-shaping can be found in [5] and [6].) In the next section we describe a computationally efficient linear-phase dc blocking network.

LINEAR-PHASE DC BLOCKER

Another approach to eliminate the dc bias of a signal is to compute the moving average of a signal and subtract that average value from the signal as shown in Figure 3(a). The delay element in the figure is a simple delay line, having a length equal to the averager's group delay, enabling us to time-synchronize the averager's $v[n]$ output with the $x[n]$ input in preparation for the subtraction operation.

The most computationally efficient form of a D-point moving averager (MA)



[FIG3] **Linear-phase dc blocker: (a) moving average subtraction method and (b) dual-MA implementation.**

is the network whose transfer function is defined by

$$H_{MA}(z) = \frac{1}{D} \cdot \frac{1-z^{-D}}{1-z^{-1}}. \quad (6)$$

The second ratio in (6) is merely a recursive running sum comprising a D-length comb filter followed by a digital integrator. If D is an integer power of two, the $1/D$ scaling in (6) can be performed using a binary right shift by $\log_2(D)$ bits.

However, if D is an integer power of two, then the MA's group delay is not an integer number of samples, making the synchronization of the delayed $x[n]$ and $v[n]$ difficult. To solve that problem we use two cascaded D-point MAs as shown in Figure 3(b). Because the dual-MA has an integer group delay of D–1 samples, our trick is to tap off the first averager's delay line, eliminating the bottom-path "Delay" element in Figure 3(a).

The magnitude response of our dual-MA dc blocker, for $D = 32$, is shown in Figure 4(a). In the figure we show the details of this dc blocker's passband with its peak-peak ripple of 0.42 dB. The frequency axis value of 0.5 corresponds to a cyclic frequency of half the input signal's F_s sample rate. This dc blocker has the desired infinite attenuation at 0 Hz.

What we've created then is a linear-phase, multiplierless, dc blocking network having a narrow transition region near 0 Hz. It's worth noting that standard tapped delay-line, linear-phase finite-impulse response (FIR) filter designs using least-squares error minimization, or the Parks-McClellan method, require more than 100 taps to approximate our $D = 32$ dc blocking filter's performance.

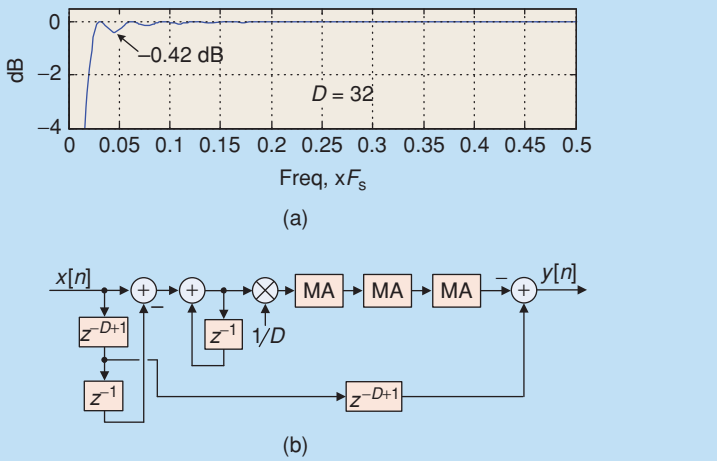
On a practical note, the MAs in Figure 3(b) contain integrators that can experience data overflow. (An integrator's gain is infinite at dc!) Using two's complement fixed-point arithmetic avoids integrator overflow errors if we ensure that the number of integrator (accumulator) bits are at least

$$\text{accumulator bits} = \text{number of bits in } q[n] + \lceil \log_2(D) \rceil, \quad (7)$$

where $q[n]$ is the input sequence to an accumulator and $\lceil k \rceil$ means as follows: if k is not an integer, round it up to the next larger integer.

For a narrower transition region width, in the vicinity of 0 Hz, than that shown in Figure 4(a), we can set D to a larger integer power of two. However, this will not reduce the dc blocker's passband ripple.

At the expense of three additional delay lines, and four new addition operations per output sample, we can implement the dc blocker shown in Figure 4(b). That quad-MA implementation, having a group delay of $2D$–2 samples, yields an improved passband peak-peak ripple of only 0.02 dB as well as a reduced-width transition region relative to the dual-MA implementation. The dc blocker in Figure 4(b) contains four $1/D$ scaling operations which, of course, can be combined and implemented as a single binary right shift by $4\log_2(D)$ bits.



[FIG4] DC blocker: (a) $D = 32$ dual-MA passband performance and (b) quad-MA implementation.

CONCLUSIONS

We presented a nonlinear-phase, but computationally efficient, dc blocking filter that achieves ideal operation when output data quantization is used. In addition, we described an alternate dc blocking filter that, at the expense of larger data memory, exhibits a linear-phase frequency response.

AUTHORS

Randy Yates (yates@ieee.org) is a consultant in audio, communications, and signal processing systems at Digital Signal Labs, Inc. He is currently pursuing an M.S. degree in electrical engineering at North Carolina State University, Raleigh, North Carolina.

Richard Lyons (R.Lyons@ieee.org) is a consulting systems engineer and lecturer with Besser Associates in Mountain View, California. He is the author of *Understanding Digital Signal Processing 2/E* (Prentice-Hall, 2004), and editor of, and contributor to, *Streamlining Digital Signal Processing, A Tricks of the Trade Guidebook* (IEEE Press/Wiley, 2007).

REFERENCES

[1] C. Dick and F. Harris, "FPGA signal processing using sigma-delta modulation," *IEEE Signal Proc. Mag.*, vol. 17, no. 1, pp. 200–35, Jan. 2000.

[2] K. Shenoi, *Digital Signal Processing in Communications Systems*. New York: Chapman & Hall, 1994, p. 275.

[3] A. Bateman, "Implementing a digital ac coupling filter," *GlobalDSP Mag.*, Feb. 2003 [Online]. Available: http://www.globaldsp.com/index.asp

[4] R. Bristow-Johnson, "DSP trick: Fixed-point dc blocking filter with noise-shaping" [Online]. Available: http://www.dspguru.com/comp.dsp/tricks/alg/dc block.htm

[5] P. Aziz, H. Sorensen, and J. Van Der Spiegel, "An overview of sigma-delta converters," *IEEE Signal Proc. Mag.*, vol. 13, no. 1, pp. 61–84, Jan. 1996.

[6] G. Bourdopoulos, A. Pnevmatikakis, V. Anastassopoulos, and T.L. Deliyannis, *Delta-Sigma Modulators: Modeling, Design and Applications*. London: Imperial College Press, 2003, pp. 36–40. SP