

A MIXED-SIGNAL APPROACH FOR TUNING CONTINUOUS-TIME LOW-PASS FILTERS

Anthony Tong and Paul J. Hurst

Solid-State Circuits Research Laboratory

Department of Electrical and Computer Engineering

University of California

Davis, CA 95616

hurst@ece.ucdavis.edu

¹This work was funded in part by UC MICRO grant 01-084, with support from Broadcom, Intel, Metalink, National Semiconductor, TDK Semiconductor and Texas Instruments.

Abstract

A mixed-signal approach for tuning the bandwidth of continuous-time low-pass filters is presented. The tuning loop uses common circuit blocks (a data converter and digital filters). Simulation results are presented for a number of filters, and measured results are presented for tuning second-order and fourth-order filters.

1. INTRODUCTION

Continuous-time low-pass filters are used in mixed-signal ICs in applications including anti-aliasing or output smoothing. The bandwidth of a continuous-time (CT) low-pass filter (LPF) is determined by time constants (such as $R_i C_i$ or g_{mi}/C_i) that are poorly controlled in an integrated-circuit process. To set the filter bandwidth accurately, the time constants in the filter can be adjusted by trimming or tuning. Trimming is undesirable because it increases test time, which increases costs, and cannot correct for changes that occur after trimming (due to temperature changes, for example). On-chip tuning typically uses extra circuitry to automatically adjust the time constants, usually by controlling transconductances, resistances, or capacitances in the filter [1], [2], [3], [4], [5]. In some on-chip tuning schemes, one or two replicas of the integrators in the filter and an accurate reference, such as an external clock or a precision external resistor, are used in a feedback loop that adjusts the time constants in the replica integrators. The control signal that sets time constants in the replica integrators is also used to set the time constants in the integrators in the filter. Advantages of this approach are that the filter is insensitive to temperature or component changes and that the tuning is done in the background (i.e., the filter can be used during tuning). However, the extra circuits for tuning require extra IC area and power. Also, since the CT filter is tuned indirectly, mismatch between the replica integrators and those in the filter limits the accuracy of the filter bandwidth.

A mixed-signal scheme for direct tuning of a CT filter based on model matching was presented by Kozma, *et al.* [6]. With this approach, a test input signal is applied to the CT filter and to an ideal model of the filter. An error is computed as the difference between the filter output and the ideal output. The error signal is multiplied by a gradient, which is generated by another analog filter, and the product is integrated to adjust each filter coefficient. With this approach, coefficients can be adapted to adjust the poles, zeros and DC gain of a filter. This tuning scheme produces an accurate transfer function, even if element matching is poor. However, it requires extra circuits to generate the gradients and the ideal filter output.

In this paper, a simpler mixed-signal approach is proposed that directly tunes only the bandwidth of a

CT LPF. With good component matching, such tuning is sufficient except when the quality factor(s) must be tuned, as may be required in very high-frequency filters or filters with high-Q poles. An analog-to-digital converter (ADC) and a digital filter are used in the tuning loop. These blocks exist on many ICs. To save IC area, it may be possible to time share these circuit blocks, allowing a CT LPF to be directly tuned whenever these circuit blocks are not processing an input signal. An example application of this tuning scheme is in mixed-signal digital communication transceivers that include a digital-signal processor (DSP). For example, when not receiving data (e.g., during transmission of a handshake sequence or after power-up), the ADC and DSP could be used to tune a CT LPF in the receiver. Specific application examples for the proposed tuning approach are the CT LPFs in mixed-signal implementations of ethernet transceivers, DSL transceivers, and disk-drive read channels. If time sharing of circuit blocks is not practical, the proposed tuning technique can be applied using replica circuits, as described in Section 2.

2. TUNING APPROACH

The filter tuning concept is shown in Fig. 1a. For simplicity here, assume that CT filter $H(s)$ is a 1-pole filter, but its time constant τ (and therefore its -3dB bandwidth $f_0 = 1/2\pi\tau$) differs from the desired value due to variations in filter component values. Also, assume that the filter time constant can be adjusted by means of a control signal.

In Fig. 1a, a digital input sequence $x[n]$ is converted to an analog signal by a digital-to-analog converter (DAC) operating at a sample rate of f_S . The DAC outputs a zero-order-held signal $x_a(t)$. The DAC output is filtered by the CT filter $H(s)$. The filter output is sampled at a rate f_S and converted to a digital signal by the ADC. (The ADC and DAC use the same clock f_S .) The ADC output samples $v[n]$ are processed digitally by the remaining blocks in Fig. 1a. The samples $v[n]$ are filtered by discrete-time filter $F(z)$ to produce its output samples $y[n]$. These samples are processed to generate a signal to tune filter $H(s)$.

To adjust the time constant of $H(s)$ in Fig. 1a, known input samples $x[n]$ are applied that produce a known (or correct) output $y[n]$ when the filter $H(s)$ has the desired time constant. As an input, a simple periodic sequence with a period of four samples is chosen where $x[-2] = x[-1] = -1$, $x[0] = x[1] = +1$ and $x[n+4] = x[n]$.

The algorithm used to tune $H(s)$ is developed in the following analysis. For simplicity, the tuning algorithm will be derived for a one-pole $H(s)$; tuning higher order filters will be considered in the next section. Because the 1-bit DAC and ADC use the same clock, the processing of $x[n]$ to produce $v[n]$ can be described by a linear filter that is the discrete-time (DT) equivalent of the signal path from $x[n]$ to $v[n]$ in Fig. 1a. This DT filter is $G(z)$ in Fig 1b [7]. The DT equivalent filter $G(z)$ is the DT filter that will

produce the same output $v[n]$ that is produced by the cascaded DAC, CT LPF, and ADC in Fig. 1a, if the same input $x[n]$ is applied to both. (The effect of ADC quantization is not included in Fig. 1b.) If $H(s)$ has one pole and no zeros, the DT equivalent transfer function from $x[n]$ to $v[n]$ also has one pole and no zeros (see Appendix). Therefore, the relationship between $x[n]$ and $v[n]$ can be expressed as

$$v[n] = pv[n - 1] + gx[n - 1] \quad (1)$$

or, equivalently, in the z domain by the transfer function

$$G(z) = \frac{V(z)}{X(z)} = \frac{gz^{-1}}{1 - pz^{-1}} \quad (2)$$

where

$$p = \exp\left(\frac{-1}{\tau f_S}\right) \quad (3)$$

is the pole of this DT equivalent filter, and g sets the filter gain. If τ_d is the desired time constant of $H(s)$, the desired value of the pole is $p_d = \exp(-1/\tau_d f_S)$. Call the desired gain value in (1) g_d . [This g_d is determined by the desired gain of the CT LPF $H(s)$.] Also, call $H_d(s)$ and $G_d(z)$ the CT and DT-equivalent transfer functions when the time constant and gain take their desired values of τ_d and g_d . (Throughout this paper, the subscript 'd' refers to the desired value of the variable.) A one-zero causal filter $F(z)$ that has a zero equal to the pole p_d in $G_d(z)$ and a gain that is the reciprocal of the desired gain g_d in $G_d(z)$ is described in the time domain by

$$y[n] = \frac{v[n] - p_d v[n - 1]}{g_d} \quad (4)$$

or in the z domain by

$$F(z) = \frac{Y(z)}{V(z)} = \frac{1 - p_d z^{-1}}{g_d} \quad (5)$$

This $F(z)$ is a causal inverse of $G_d(z)$. When $H(s)$ has the desired gain and time constant τ_d , the pole in $G(z)$ and the zero in $F(z)$ are both equal to p_d ; therefore, $G(z)F(z) = z^{-1}$ and $y[n] = x[n - 1]$. Deviations of the gain and/or time constant of $H(s)$ from the desired values will cause $y[n] \neq x[n - 1]$. The goal is to generate a signal that is a function of the error in the filter time constant or pole and that can be used to tune the filter. One such signal is derived next.

Replacing $v[n]$ in (4) with (1) and rearranging gives

$$y[n] = \frac{g}{g_d} x[n - 1] + \frac{(p - p_d)}{g_d} v[n - 1] \quad (6)$$

From (6), the difference $y[n + 1] - y[n]$ can be written as

$$y[n + 1] - y[n] = \frac{g}{g_d} (x[n] - x[n - 1]) + \frac{(p - p_d)}{g_d} (v[n] - v[n - 1]) . \quad (7)$$

Consecutive input samples equal 1 once every 4 samples; i.e., $x[4m + 1] = x[4m] = 1$. Setting $n = 4m + 1$ in (7) and using $x[4m + 1] = x[4m] = 1$ gives

$$y[4m + 2] - y[4m + 1] = \frac{(p - p_d)}{g_d} (v[4m + 1] - v[4m]) . \quad (8)$$

The samples $x[4m + 1] = x[4m] = 1$ make the CT signal $x_a(t)$, which is the input to $H(s)$, a constant positive signal for two sample periods, as shown in Fig. 2a. During these two periods, for any 1-pole $H(s)$, its CT output $v_a(t)$ is strictly increasing with time as shown in Fig. 2a. Therefore, the corresponding two samples of $v_a(t)$ are increasing with time, so $v[4m + 1] > v[4m]$, or $v[4m + 1] - v[4m] > 0$. Calling the difference on the left side of (8) the error $e[m]$, we have

$$e[m] = y[4m + 2] - y[4m + 1] \quad (9)$$

$$= (p - p_d) w[m] \quad (10)$$

where $w[m] = (v[4m + 1] - v[4m]) / g_d$. In the following, we will assume $g_d > 0$. [The case $g_d < 0$ can be handled by making μ negative in (11).] With $g_d > 0$, $w[m] > 0$. Therefore, the error $e[m] > 0$ if $p > p_d$ (or, equivalently, if $\tau > \tau_d$); $e[m] < 0$ when $p < p_d$ (or $\tau < \tau_d$); and $e[m] = 0$ when $p = p_d$ (or $\tau = \tau_d$). Thus, $e[m]$ is a useful detector of error in the pole or time constant of the one-pole filter $H(s)$. Using $e[m]$, the filter time constant, which is inversely related to the bandwidth, can be adjusted by accumulating the error

$$\tau[m + 1] = \tau[m] - \mu e[m] \quad (11)$$

where μ is a positive scale factor. When (11) is used to adjust the time constant of $H(s)$, a negative feedback loop is formed. The time constant is adjusted by scaling elements in the filter. In steady state, the negative feedback forces the average value of the input $e[m]$ of the accumulator in (11) to zero, and τ converges to τ_d . Therefore, the filter $H(s)$ has the desired time constant (and bandwidth) in steady state.

Fig. 2 shows three examples of the waveforms in Fig. 1a. In all cases, the desired filter bandwidth $f_{0d} = 1/2\pi\tau_d = 1$ kHz, and the sampling frequency $f_S = 2$ kHz. The choice of $f_S/2 \approx f_{0d}$ is reasonable, since the LPF being tuned might be the anti-alias filter for the ADC when the filter and ADC are processing an input signal. Fig. 2a, b and c show the waveforms when the CT filter has the desired bandwidth (i.e., $f_0 = f_{0d}$), a bandwidth less than desired (i.e., $f_0 < f_{0d}$), and a bandwidth greater than

desired (i.e., $f_0 > f_{0d}$), respectively. In each case, the top plot shows the input samples $x[n]$ and the DAC output $x_a(t)$. The next plot shows the continuous-time output $v_a(t)$ of filter $H(s)$, and the samples $v[n]$ are marked on the $v_a(t)$ waveform. The bottom plot shows the output $y[n]$ of the finite-impulse-response (FIR) filter $F(z)$ in (5). From (9), $e[m] = 0$, $e[m] > 0$, and $e[m] < 0$, respectively, in these three cases.

Intuitively, the filter in Fig. 1a is tuned when the output samples $y[n]$ are a delayed version of the input samples $x[n]$. Rather than using all four samples of $y[n]$ each period, the error is computed here using only the two positive samples of $y[n]$. If desired, the two negative samples each period could be used to compute an error (based on their difference) and used in addition to, or instead of, the error in (9). While this calculation would be relatively simple, it is not covered in this paper for simplicity.

Fig. 3 shows a plot of the error $e[m]$ from (9) plotted versus filter bandwidth $f_0 = 1/2\pi\tau$ for a desired filter bandwidth f_{0d} of 1 kHz. From simulation, the error $e[m]$ is zero when $f_0 = f_{0d}$, $e[m]$ is positive for $0 < f_0 < f_{0d}$, and $e[m]$ is negative for $f_0 > f_{0d}$. When the filter time constant is adjusted recursively using (11), the one-pole CT filter will converge in steady state to the desired bandwidth f_{0d} from any initial bandwidth f_0 .

In practice, the filter time constant is controlled by changing a digital word that controls switches in an array of elements [8], [9] or by changing a dc voltage or current that varies element value(s) [10].

There are many advantages of this tuning scheme. First, the error signal $e[m]$ is not affected by offset in the analog circuits (the DAC, CT filter or ADC in Fig. 1a) because $e[m]$ is the difference between consecutive samples of the ADC output. Second, variation of the dc gain of the CT filter has little impact on the tuning loop. The expected dc gain of the CT filter is used to calculate g_d in $F(z)$ in (5). If the actual CT filter gain is not equal to the desired gain g_d , the magnitude of the error $e[m]$ in (9) is scaled by the actual filter gain divided by the desired filter gain. However, this scaling does not change the steady-state operating point of the tuning loop. (Note that gain errors in the DAC and ADC can be lumped into the filter gain; therefore, small gain errors in the data converters are unimportant.) Also, the filter $F(z)$ and the differencing in (9) can be combined into a second-order difference equation, which could be implemented using an existing FIR filter (e.g., using the FIR equalization filter in a digital-communication receiver when data is not being received).

A disadvantage of this approach is that it is a foreground tuning method. Therefore, the filter cannot be used to process input signals when it is being tuned. This may be acceptable in some applications. For example, the receive filter in a digital-communication receiver could be tuned when not processing a receive signal (e.g., during transmission of a handshake sequence or after power-up of the receiver). If

foreground tuning is not acceptable, continuous background tuning could be implemented at the expense of additional circuitry by using a master-slave tuning scheme. The system in Fig. 1a could operate as the master tuning circuit, and the tuning signal generated there would also be used to tune the slave filter, which could be a replica of the master filter. The slave filter could continuously operate on an input signal. However, mismatch between elements in the master and slave filters would limit the accuracy of the bandwidth of the slave filter, and components dedicated to tuning, such as an ADC and FIR filter, would be needed. Another way to apply this tuning scheme in the background would be to have two identical CT LPFs that alternate between processing the input signal and being tuned [11].

3. Tuning Higher Order Filters

The filter tuning scheme shown in Fig. 1a and described in the last section works for any one-pole CT filter. To apply this tuning scheme to a higher order filter $H(s)$, the filter must have time constants that can be adjusted through a single control signal. Also, the filter must be designed so that the control signal changes the filter bandwidth without changing the filter in any other way. In other words, the control signal should only frequency scale the filter transfer function. This can be accomplished if the control signal scales a set of similar elements (i.e, all resistances, all transconductances, or all capacitances) by the same factor because the quality factors associated with the filter poles are dependent on ratios of like components and therefore are usually not tuned [12].

As an example, consider the two-pole filter shown in Fig. 4. Here, the value R_1 of the four matched resistors is varied by changing a time-constant control signal. In this filter,

$$f_0 \propto 1/R_1 \quad (12)$$

$$Q = \sqrt{C_2/C_1}. \quad (13)$$

Therefore, changing R_1 will change f_0 , which determines the filter bandwidth, but will not change Q , the quality factor.

If $H(s)$ in Fig. 1a has k poles, the DT equivalent transfer function $G(z)$ in Fig. 1b has k poles and may also have zeros if $k > 1$. Therefore, in general, the inverse of the DT equivalent filter would have k zeros and may also have one or more poles. Such an inverse filter, could be used for $F(z)$. However, for any $H(s)$, a simpler filter $F(z)$ with only one zero can be found that will produce an output $y[n]$ that is a delayed version of the input $x[n]$ when the periodic sequence $1, 1, -1, -1, \dots$ is applied in Fig. 1a. This input sequence can be expressed as

$$x[n] = \sqrt{2}\cos(2\pi t_n f_s/4 - \pi/4) \quad (14)$$

where $t_n = n/f_S$. Thus, $x[n]$ is a sinusoid at frequency $f_S/4$ sampled at a rate f_S . Since the input to the DT equivalent filter $G(z)$ in Fig. 1b is samples of a sinusoid, the filter output $v[n]$ is also samples of a sinusoid. Therefore, a one-zero FIR filter

$$F(z) = a + bz^{-1} \quad (15)$$

can generate an output $y[n]$ that equals the input $x[n]$ delayed because the two coefficients a and b can be chosen to compensate the magnitude and phase changes introduced by $G(z)$ at the signal frequency. The two coefficients of $F(z)$ in (15) can be found using equations for $y[n]$ at consecutive sample times n and $n + 1$:

$$\begin{aligned} y[n] &= av[n] + bv[n-1] \\ y[n+1] &= av[n+1] + bv[n] \end{aligned} \quad (16)$$

The desired output values $y[n] = y[n+1] = 1$ and the corresponding samples $v[n]$ of the desired CT filter output (determined by calculation or simulation) can be substituted in (16) to find the coefficients of $F(z)$. With this $F(z)$, $y[n]$ will be a delayed version of $x[n]$ when $H(s)$ has the desired bandwidth. Therefore, two consecutive samples of $y[n]$ in every period of four samples will be equal and positive when the CT filter has the desired bandwidth, and the difference between these samples in (9) gives the error $e[m]$, which equals zero when the filter $H(s)$ has the desired bandwidth.

For example, Fig. 5 shows the CT filter output $v_a(t)$ and the samples $v[n]$ for a two-pole Butterworth LPF with $f_{0d} = 1$ kHz and $f_S = 2$ kHz. Using $v[n-1] = -1.029$, $v[n] = 0.958$, $v[n+1] = 1.029$ from simulation and $y[n] = y[n+1] = 1$ in (16) gives $a = 1.00526$ and $b = -0.0359$.

Next, an analysis is carried out to show that the error in (9), when used in an expression like (11), can be successfully used to tune the bandwidth of a higher order filter. First, the periodic input sequence $x[n]$ in Fig. 1 can be expressed as a sampled sinusoid, as in (14). Since $x[n]$ in Fig. 1b is filtered by $G(z)$ and $F(z)$ to give $y[n]$, $y[n]$ is also a sampled sinusoid:

$$y[n] = |F(z_i)||G(z_i)|\sqrt{2}\cos[\pi n/2 - \pi/4 + \angle G(z_i) + \angle F(z_i)] \quad (17)$$

where $z_i = \exp(j\pi/2)$ is the value of z corresponding to an input frequency of $f_S/4$. Using (9), the error is

$$\begin{aligned} e[m] &= y[4m+2] - y[4m+1] \\ &= |F(z_i)||G(z_i)|\sqrt{2}\{\cos[\pi(4m+2)/2 - \pi/4 + \angle G(z_i) + \angle F(z_i)] \\ &\quad - \cos[\pi(4m+1)/2 - \pi/4 + \angle G(z_i) + \angle F(z_i)]\} \end{aligned} \quad (18)$$

Using $\cos(x - y) - \cos(x + y) = 2\sin(x)\sin(y)$ and $\sin(\pi/4) = 1$, (18) can be rewritten as

$$\begin{aligned} e[m] &= -2|F(z_i)||G(z_i)|\sqrt{2}\sin[2\pi m + \pi/2 + \angle G(z_i) + \angle F(z_i)] \\ &= -2|F(z_i)||G(z_i)|\sqrt{2}\sin[\pi/2 + \angle G(z_i) + \angle F(z_i)] \end{aligned} \quad (19)$$

To give $e[m] = 0$, $F(z)$ is designed [see (16)] to give $\angle G(z_i) + \angle F(z_i) = -\pi/2$ when the CT filter has the desired bandwidth f_{0d} [i.e., when $H(s) = H_d(s)$ and $G(z) = G_d(z)$]. (When the input, which has a period of 4 samples, experiences a phase shift of $-\pi/2$ passing through F and G , the output $y[n]$ of the FIR filter will be equal to the input $x[n]$ delayed by one sample, as desired.) Therefore, $\angle F(z_i)$ can be written as

$$\angle F(z_i) = -\pi/2 - \angle G_d(z_i) \quad (20)$$

Substituting (20) into (19) gives

$$e[m] = -2|F(z_i)||G(z_i)|\sqrt{2}\sin[\angle G(z_i) - \angle G_d(z_i)] \quad (21)$$

From this equation, it can be seen that $e[m] = 0$ when $f_0 = f_{0d}$. When f_0 changes, $H(s)$ and thus $G(z_i)$ change; therefore, the error changes.

From (21), the tuning range for a CT filter can be determined. In (21), $G(z_i)$ is a function of the filter bandwidth f_0 . The coefficients of $F(z)$ are chosen so that the error $e[m]$ is zero when the filter bandwidth is the desired value f_{0d} . Using (11), negative feedback will cause the tuning loop to converge to the desired bandwidth f_{0d} from any initial bandwidth if the error is positive for $f_0 < f_{0d}$ and negative for $f_0 > f_{0d}$, as in the plot for the one-pole filter in Fig. 3. [A sign reversal of the error can be handled by letting $\mu \rightarrow -\mu$ in (11).] However, for high-order filters, the error may equal zero for one or more bandwidths other than f_{0d} . This will occur if there is a bandwidth f_0 for which $\angle G(z_i)$ changes by π radians (or an integer multiple thereof) from the value $\angle G_d(z_i)$. With such phase shift, $\sin(\cdot) = 0$ in (21), which gives $e[m] = 0$. If the error is zero for multiple filter bandwidths, the tuning loop will converge to the desired bandwidth f_{0d} only from a range of bandwidths around f_{0d} that is bounded by frequencies where the error e goes to zero.

Using (9) to generate the error signal, Fig. 6 shows the simulated plots of error $e[m]$ vs. bandwidth f_0 for Butterworth LPFs of order 2, 4 and 6. For each filter, the coefficients of $F(z)$ were calculated for a desired filter bandwidth of $f_{0d} = 1$ kHz. The error $e[m] = 0$ when $f_0 = f_{0d}$, $e[m] < 0$ for $f_0 > f_{0d}$, and $e[m] > 0$ for a large range of bandwidths less than f_{0d} . When the filter time-constant control signal is adjusted recursively using (11) from any initial bandwidth below f_{0d} where $e[m] > 0$ or any initial

bandwidth above f_{0d} , the CT filter will converge in steady state to the desired bandwidth. (The multiple zeros of the error for the fourth- and sixth-order filters are due to a phase change that exceeds π radians in magnitude as the bandwidth decreases well below f_{0d} , as described above.) These curves show that the proposed tuning method can be used to tune these Butterworth filters from initial bandwidths that are in a range greater than $\pm 50\%$ of the desired filter bandwidth.

Simulations show that the tuning technique also works for Bessel filters of order 2-8 over a frequency range of at least $\pm 50\%$ of the desired bandwidth. The error vs. bandwidth curves for Bessel filters of order 2, 4 and 6 are shown in Fig. 7. In these simulations, the desired 3-dB bandwidth is $f_{0d} = 900$ Hz and $f_s = 2$ kHz. Also from simulation, the tuning method works over a frequency range of at least $\pm 50\%$ for a second-order elliptic LPF with 2-dB passband ripple and 60-dB stop-band rejection.

For any CT LPF, the error vs. bandwidth plot can be generated from simulation. Such simulation could be carried out using (9) and Fig. 1a or 1b. However, using Fig. 1b may require significant effort to find $G(z)$ as a function of bandwidth f_0 . Therefore, for the simulations in this paper, Fig. 1a was simulated in MATLAB with $H(s)$ modeled by a DT filter with an impulse response equal to the impulse response $h(t)$ highly oversampled, with a sampling rate of $100f_s$. As noted above, if the error is zero for multiple filter bandwidths, the tuning loop will converge to the desired bandwidth f_{0d} only from a range of bandwidths around f_{0d} that is bounded by bandwidths where the error goes to zero. Based on simulations, the tuning range is sufficiently large for some popular CT LPFs. However, if the error plot does not indicate a sufficiently large tuning range, then the master/slave version of the tuning scheme described in the last section could be used, with the master consisting of a tunable low-order filter that uses components related to those in the high-order slave filter. Alternatively, if the CT filter consists of cascaded biquads, each biquad could be tuned separately. The limited phase shift in a two-pole filter assures that the tuning loop converges from any initial bandwidth, as can be seen for the second-order plots in Figs. 7 and 8.

3.1 Limitations

The proposed tuning scheme requires good matching of similar elements in the filter, so that a single control signal can scale all the time constants together by scaling all like elements by the same factor. If the matching is not good, coefficients that determine the filter transfer function could be adapted using a least-mean squared (LMS) approach [6], which does not rely on matching. However, the LMS approach is more complex than the proposed tuning method. With LMS tuning, multiple filter coefficients would be adapted, but a gradient value would be needed for each LMS coefficient update, requiring the addition

of gradient filter(s) [6].

Signal quantization by the ADC in Fig. 1a can affect the tuning accuracy. Quantization 'noise' is introduced by the ADC, and this noise $q[n]$ can be modeled as a signal added to the input of filter $F(z)$ in Fig. 1a. The component of the error e due to this quantization noise, which will be called $e_q[n]$, can be found by considering $q[n]$ to be the only input to filter $F(z)$. In this case, the output of the filter $F(z)$ is [from (15)]

$$y[n] = f[n] * q[n] = aq[n] + bq[n - 1] \quad (22)$$

where '*' denotes convolution. Using (9) and (22), $e_q[n]$, the component of the error due to $q[n]$, is given by

$$e_q[m] = y[4m + 2] - y[4m + 1] = aq[4m + 2] + (b - a)q[4m + 1] - bq[4m] \quad (23)$$

This e_q will have its largest impact on the tuning loop under the following conditions. If the quantization noise is periodic with a period of 4 samples, $e_q[m]$ will be a dc signal and will contribute a dc offset at the accumulator input. Since the average accumulator input in Fig. 1a must be zero in steady state, the dc component of the error e due to the signal x (ignoring quantization noise $q[n]$) will have to equal $-e_q$. Therefore, nonzero e_q will result in a misadjustment of the filter bandwidth. The amount of misadjustment depends on the value of e_q . If each sample of q takes its worst-case magnitude of 1/2 LSB and has a sign that causes all the terms in (23) to combine constructively, the largest magnitude of e_q is

$$\max|e_q[n]| = (|a| + |b - a| + |b|)(1/2 \text{ LSB}) \quad (24)$$

For example, consider the second-order Butterworth filter considered in Section 3. For that filter, the coefficients of $F(z)$ are $a = 1.00526$ and $b = -0.0359$, and (24) gives $\max|e_q[n]| = 2.082(1/2 \text{ LSB})$. With an 8-bit ADC with a normalized full scale of ± 1.1 , 1/2 LSB is 0.0043, and $\max|e_q| = 0.0089$. With an error of 0.0089, a zoomed-in view of the plot in Fig. 6 near f_{0d} shows that a bandwidth shift of about 9 Hz results. This is a worst-case error of 0.9% in the filter bandwidth, due to 8-bit ADC quantization.

Typically, the tuning error will be smaller than predicted by the worst-case analysis for multiple reasons. First, it is unlikely that three consecutive quantization values will all be 1/2 LSB in magnitude. For example, simulations of Fig. 1a with an 8-bit ADC for this second-order Butterworth filter gave shifts in the filter bandwidth after tuning of less than 4.5 Hz from $f_{0d} = 1 \text{ kHz}$, or 0.45% of f_{0d} . Also, the CT filter and the ADC itself introduce circuit noise in the ADC input $v_a(t)$ that acts as dither. With sufficient dither, the effect of ADC quantization becomes irrelevant due to the averaging effect of the accumulator in Fig. 1 [13]. For example, running the same simulations mentioned above (in this paragraph) but adding

1 LSB of uniformly distributed noise at the ADC input, which acts as a dither signal, gave shifts in the filter bandwidth after tuning of less than 0.43 Hz, or 0.043% of f_{0d} . Therefore, the tuning is nearly perfect with 1 LSB of added dither. With 1/8 LSB of uniform dither, simulations gave shifts in the bandwidth of less than 2.2 Hz, or 0.22% of f_{0d} . So even a small amount of dither is beneficial.

The size of the worst-case shift of the filter bandwidth due to ADC quantization depends on the coefficients of $F(z)$, the number of ADC bits, and the $e[m]$ vs. bandwidth curve for the CT filter being tuned. In many communication applications, a bandwidth accuracy of several percent (e.g., $\pm 5\%$) is sufficient [9],[14],[15] because the channel significantly attenuates high frequencies. Therefore, a small change in the bandwidth of a filter that is in series with the channel has little effect on the signal and system performance.

If the processing in Fig. 1a were carried out entirely in the analog domain, using switched-capacitor circuits to implement the FIR filter $F(z)$, the differencing to compute the error in (9), and the accumulator; an ADC would not be needed. Hence, ADC quantization would not be a concern. However, these switched-capacitor circuits would introduce an offset at the accumulator input, which would cause some shift in the filter bandwidth.

Jitter in the ADC sampling clock will affect the sample value $v[n]$, which will affect the error signal $e[m]$. For small sample-clock jitter j_n , the sample value can be approximately written as

$$v[n] = v_a(nT + j_n) \approx v_a(nT) + j_n \left. \frac{dv_a(t)}{dt} \right|_{t=nT} \quad (25)$$

The first term on the right is the ideal sample value; the second term is the error in the sample value due to the jitter j_n . If the jitter is random, zero mean and uncorrelated with the signal $v_a(t)$, this second term is zero-mean noise that is added to the desired ADC output. This will introduce noise in the error $e[m]$, which will have little impact on the tuning loop due to the averaging effect of the accumulator in Fig. 1.

4. MEASURED RESULTS

A breadboard version of the tuning system in Fig. 1a was constructed. First, the tuning scheme was tested with a second-order Butterworth filter $H(s)$. This filter is implemented as an active RC LPF as shown in Fig. 4. Each variable resistor R_1 consists of a fixed resistor R in series with a MOS transistor operating in the triode region [10]. These resistors and MOS transistors are matched. The value of R_1 is varied by changing a tuning voltage, which is the gate voltage of the MOS transistors. Changing the gate voltage changes the on-resistance of each transistor. While this implementation of a variable resistor is used here in a single-ended circuit for simplicity, better filter linearity can be achieved by using these

variable resistors in a fully differential filter. The filter coefficients for $F(z)$ were computed for a desired bandwidth $f_{0d} = 1$ kHz. The FIR filter $F(z)$, the error calculation in (9), and the update of the tuning voltage are implemented in an Altera Flex 10K chip. An 8-bit ADC generates the samples $v[n]$. The sample rate is $f_S = 2$ kHz. On the breadboard, the signal $x_a(t)$ was simply generated using a flip-flop connected to clean supplies, rather than a 1-bit DAC. An 8-bit DAC converts the digital control signal (computed in the Altera chip) to an analog voltage that drives the gates of the MOS transistors.

Magnitude responses before and after tuning are shown in Fig. 8 for this second-order Butterworth filter. The filter capacitors were changed to generate five different initial conditions for tuning. The first five rows of Table 1 give the filter bandwidths before and after tuning for this filter.

Data were also gathered for a fourth-order Butterworth filter. The coefficients of $F(z)$ were again computed for a desired filter bandwidth of 1 kHz. The last 4 rows of Table 1 give the measured filter bandwidths before and after tuning. For both the second- and fourth-order filters, the filter bandwidths are within $\pm 0.3\%$ of the desired value after tuning. The tuning here benefited from the presence of circuit noise at the input of the ADC from the ADC itself and the CT filter; this noise acted as dither.

A breadboard version of the second-order elliptic filter described in the previous section was also tuned successfully. To verify through measurement that the steady-state operating point of the tuning loop is insensitive to dc offset and gain errors in the analog signal path, a gain error of 20% was introduced in the filter and then a dc offset equal to 20% of the ADC full-scale input was added to the ADC input. In both cases, the tuned bandwidth was not affected.

5. CONCLUSION

A mixed-signal approach to tuning CT low-pass filters has been described. In the simulations and measurements, the desired bandwidths were 1 kHz or less. However, this tuning technique is general and can be used for a wide range of desired bandwidths. Since the proposed tuning scheme only tunes the filter bandwidth, it is not appropriate if the quality factor(s) must be tuned, as required in some very high-frequency filters or filters with high-Q poles. For this tuning scheme to be useful, a plot of the error versus filter bandwidth must show a sufficiently wide range of bandwidths over which the tuning loop converges to the desired bandwidth, which is a range of bandwidths around f_{0d} that is bounded by bandwidths where the error goes to zero. Simulations show that the proposed method can tune Bessel and Butterworth filters at least up to order six from initial bandwidths that are in a range greater than $\pm 50\%$ of the desired filter bandwidth.

Since the filter is included in the tuning loop, this tuning scheme is a foreground tuning scheme, so

tuning must occur when the filter is not processing an input. This approach is particularly well suited to mixed signal ICs such as digital-communication transceivers that use CT filters and that contain an ADC and a DSP that can be used to form the tuning loop in Fig. 1 during power-up or when the receiver is idle. If foreground tuning is not possible, this tuning approach could be applied in the background using replica circuits as described in the paper.

Appendix

The discrete-time equivalent transfer function $G(z)$ for the DAC and one-pole $H(s)$ in series in Fig. 1a is derived in this appendix. The discrete-time equivalent impulse response $g[n]$ can be found by applying a DT delta function $\delta[n]$ as the input $x[n]$ in Fig. 1a and then finding the output sequence $v[n]$ that results (assuming an ideal ADC). This output sequence will equal the DT equivalent impulse response $g[n]$. Then $G(z)$ is found by taking the z-transform of $g[n]$.

If $x[n] = \delta[n]$, the DAC output is $p_T(t)$, a pulse of width $T = 1/f_S$. So

$$v_a(t) = p_T(t) * h(t) \quad (26)$$

where $h(t)$ is the impulse response of $H(s)$ and $*$ denotes convolution. Since a pulse can be expressed as a difference of unit steps [i.e., $p_T(t) = u(t) - u(t - T)$], $v_a(t)$ can be expressed as a difference in step responses:

$$v_a(t) = s(t) - s(t - T) \quad (27)$$

where

$$s(t) = A[1 - \exp(-t/\tau)]. \quad (28)$$

Here A is the DC gain of the LPF $H(s)$. The ADC output $v[n]$ is $v_a(t)$ sampled every $T = 1/f_S$:

$$v[n] = v_a(nT) = s(nT) - s(nT - T) \quad (29)$$

Substituting (28) into (29) and taking the z-transform of $v[n]$ gives

$$V(z) = \frac{A(1-p)z^{-1}}{1-pz^{-1}} \quad (30)$$

where $p = \exp(-1/\tau f_S)$. Since the input was $\delta[n]$, this $V(z)$ is equal to the DT equivalent transfer function $G(z)$:

$$G(z) = V(z) = \frac{gz^{-1}}{1-pz^{-1}} \quad (31)$$

where $g = A(1-p)$.

A similar analysis could be carried out to find the DT equivalent $G(z)$ for higher order $H(s)$.

REFERENCES

- [1] Y. P. Tsividis, M. Banu and J. Khoury, "Continuous-Time MOSFET-C Filters in VLSI," *IEEE J. Solid-State Circuits*, pp. 15-30, Feb. 1986.
- [2] R. Schaumann and M. A. Tan, "The Problem of On-Chip Automatic Tuning in Continuous-Time Integrated Filters," *Proc. of ISCAS*, pp. 106-109, 1989.
- [3] R. Schaumann, "Continuous-Time Integrated Filters - A Tutorial," *Proc. of IEE*, Pt. G, vol. 136, pp. 184-190, Aug. 1989.
- [4] J. Voorman, "Continuous-Time Analog Integrated Filters," in *Integrated Continuous-Time Filters*, edited by Y. P. Tsividis and J. O. Voorman, pp. 15-46, New York, IEEE Press, 1993.
- [5] Y. P. Tsividis, "Integrated Continuous-Time Filters Design - An Overview," *IEEE J. Solid-State Circuits*, pp. 166-176, March 1994.
- [6] K. A. Kozma, D. A. Johns and A. S. Sedra, "Automatic Tuning of Continuous-Time Integrated Filters Using an Adaptive Filter Technique," *IEEE Trans. on Circuits and Systems*, pp. 1241-1248, Nov. 1991.
- [7] E. A. Lee and D. G. Messerschmitt, *Digital Communication*, p. 322, Boston, Kluwer, 1988.
- [8] A. M. Durham, W. Redman-White and J. B. Hughes, "Low Distortion VLSI Compatible Self-Tuned Continuous-Time Monolithic Filters," *Proc. of ISCAS*, pp. 1448-1451, 1991.
- [9] J. B. Hughes, N. C. Bird and R. S. Sohn, "Self-Tuned RC-Active Filters for VLSI," *El. Letters*, pp. 993-994, Sept. 1986.
- [10] U-K. Moon and B-S. Song, "Design of a low-distortion 22-kHz fifth-order Bessel filter," *IEEE Journal of Solid-State Circuits*, vol. 28, pp. 1254 - 1264, Dec. 1993.
- [11] Y. P. Tsividis, "Self-Tuned Filters," *El. Letters*, pp. 406-407, June 1981.
- [12] Y. P. Tsividis and J. O. Voorman, *Integrated Continuous-Time Filters*, New York, IEEE Press, p. 389, 1993.
- [13] J. Vanderkooy and S. P. Lipshitz, "Resolution Below the Least-Significant Bit in Digital Systems with Dither," *Journal of Audio Eng. Society*, vol. 32, pp. 106-113, Mar. 1984.
- [14] C. A. Laber and P. R. Gray, "A 20-MHz Sixth-Order BiCMOS Parasitic-Insensitive Continuous-Time Filter and Second-Order Equalizer Optimized for Disk-Drive Read Channels," *IEEE Journal of Solid-State Circuits*, vol. 28, pp. 462 - 470, Apr. 1993.
- [15] G. A. De Veirman and R. G. Yamasaki, "Monolithic 10-30MHz Tunable Bipolar Bessel Lowpass Filter," *Proc. of ISCAS*, pp. 1444-1447, 1991.

Figure Captions

Fig. 1: a) Block diagram of the tuning scheme, b) simplified block diagram using the discrete-time equivalent filter $G(z)$.

Fig. 2: Signals in Fig. 1(a) for a one-pole CT LPF when: a) $H(s)$ has the desired bandwidth ($f_0 = f_{0d}$), b) the bandwidth of $H(s)$ is less than the desired value ($f_0 < f_{0d}$), and c) the bandwidth of $H(s)$ is greater than the desired value ($f_0 > f_{0d}$).

Fig. 3: Plot of error e vs. bandwidth f_0 for a one-pole $H(s)$. The desired bandwidth is 1 kHz and $f_S = 2$ kHz.

Fig. 4: A tunable two-pole filter.

Fig. 5: Output of a second-order Butterworth filter in Fig. 1(a) with $f_0 = 1$ kHz and $f_S = 2$ kHz.

Fig. 6: Plots of error e vs. bandwidth f_0 for Butterworth filters of order 2, 4 and 6 with $f_{0d} = 1$ kHz and $f_S = 2$ kHz.

Fig. 7: Plots of error e vs. bandwidth f_0 for Bessel filters of order 2, 4 and 6 with $f_{0d} = 900$ Hz and $f_S = 2$ kHz.

Fig. 8: Plots of the magnitude response for a second-order Butterworth filter before and after tuning. Five cases are shown.

TABLE I

TABLE 1. MEASURED RESULTS OF TUNING BUTTERWORTH LPFS.

Filter Order	Untuned f_0 (Hz)	Tuned f_0 (Hz)	% error in tuned f_0
2nd	850	1003	0.3 %
2nd	860	998	-0.2 %
2nd	873	1002	0.2 %
2nd	1050	1003	0.3 %
2nd	1058	998	-0.2 %
4th	855	998	-0.2 %
4th	863	1000	0.0 %
4th	1055	998	-0.2 %
4th	1060	1003	0.3 %

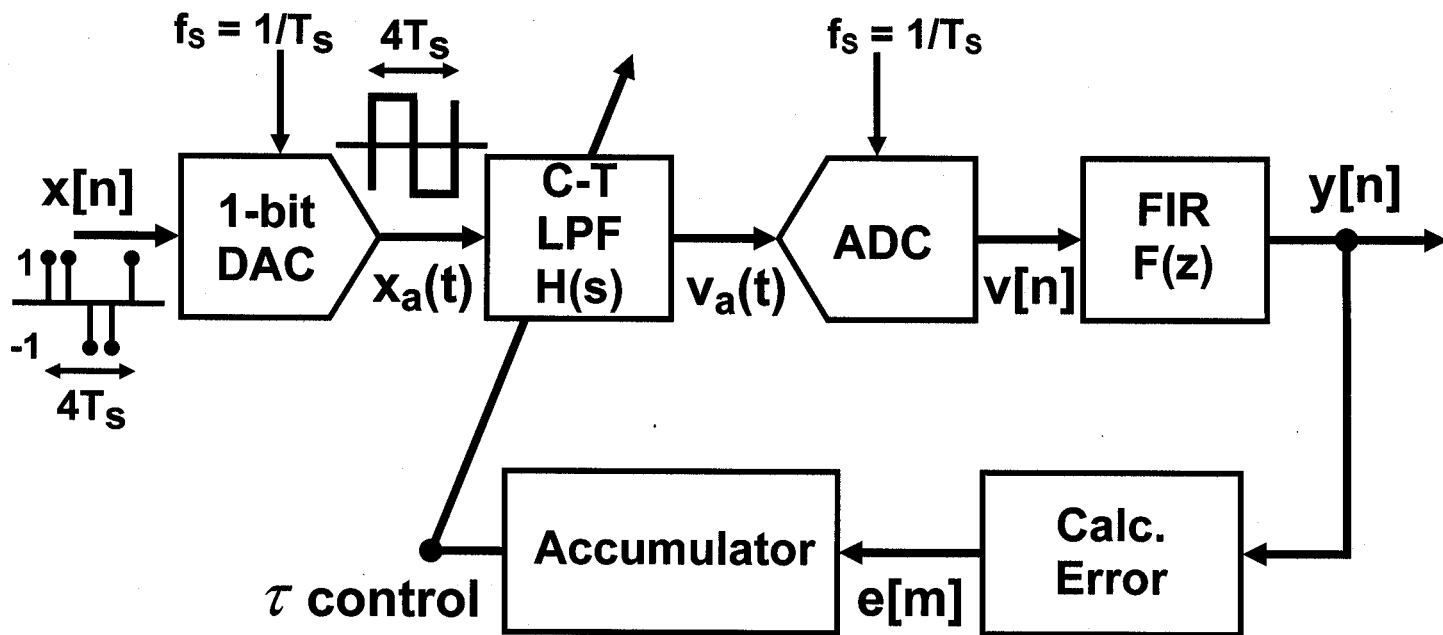


Fig. 1(a)

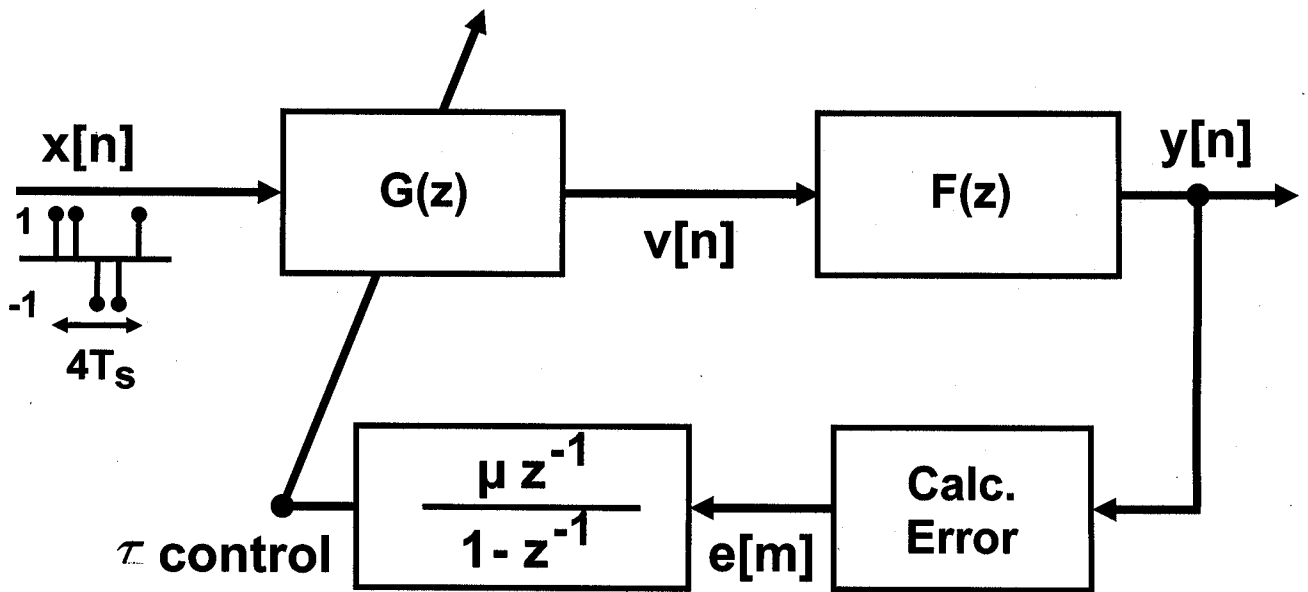


Fig. 1(b)

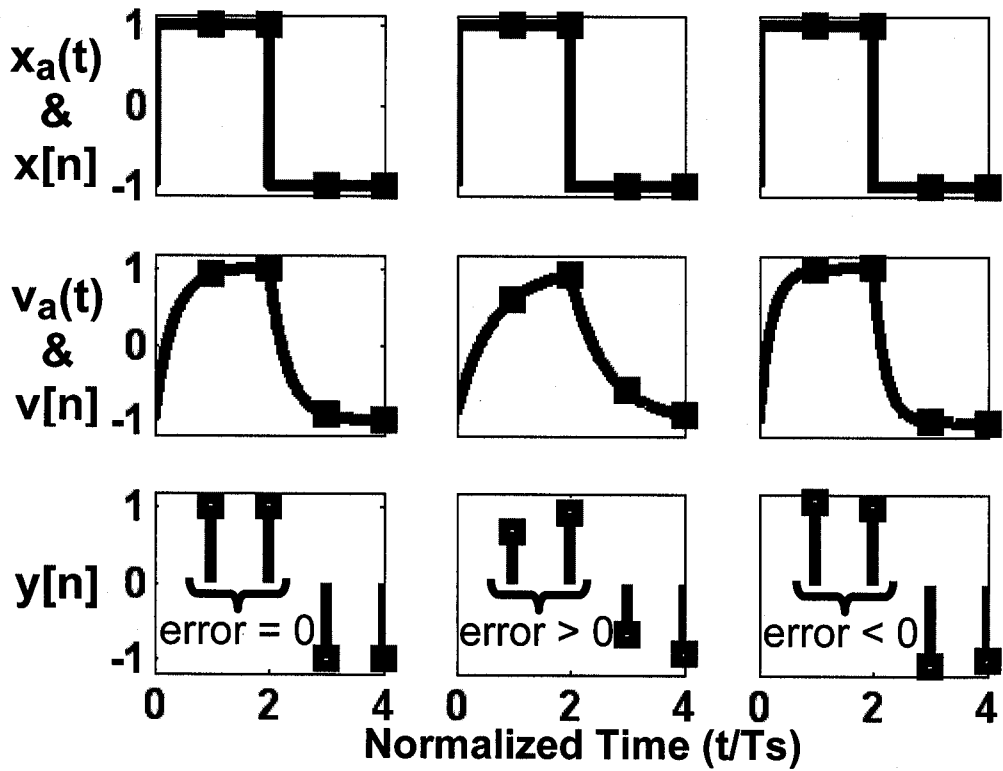


Fig. 2(a) Fig. 2(b) Fig. 2(c)

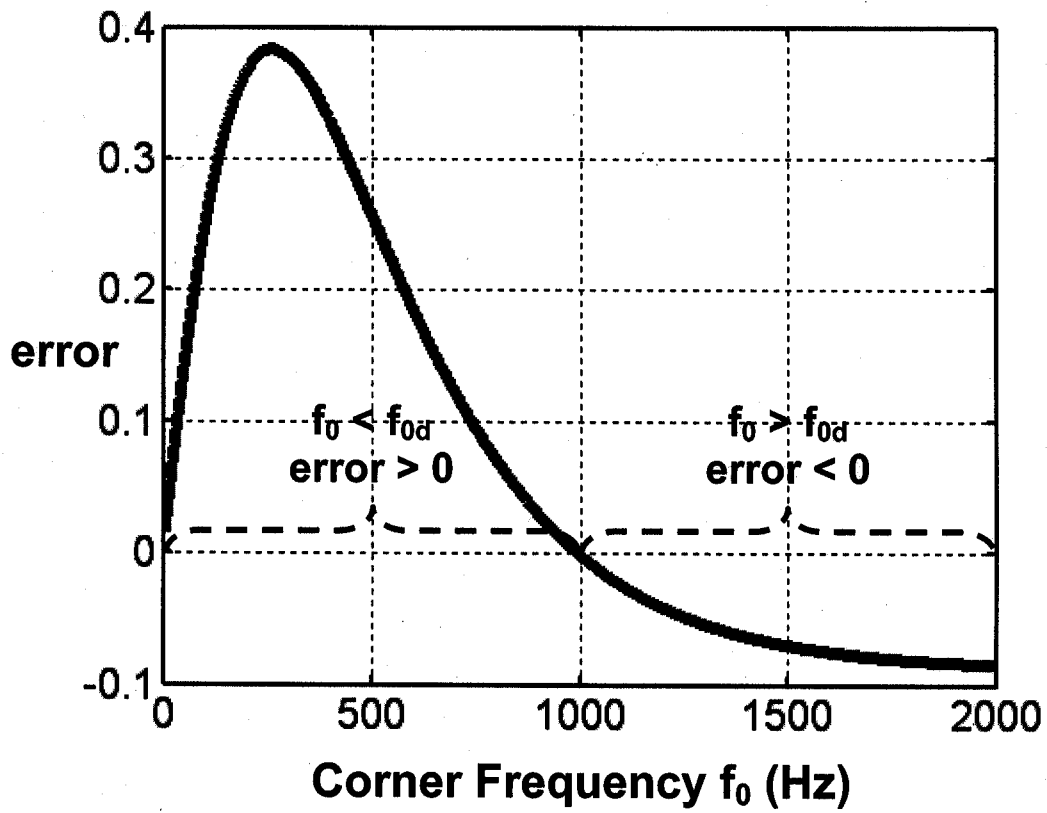


Fig. 3

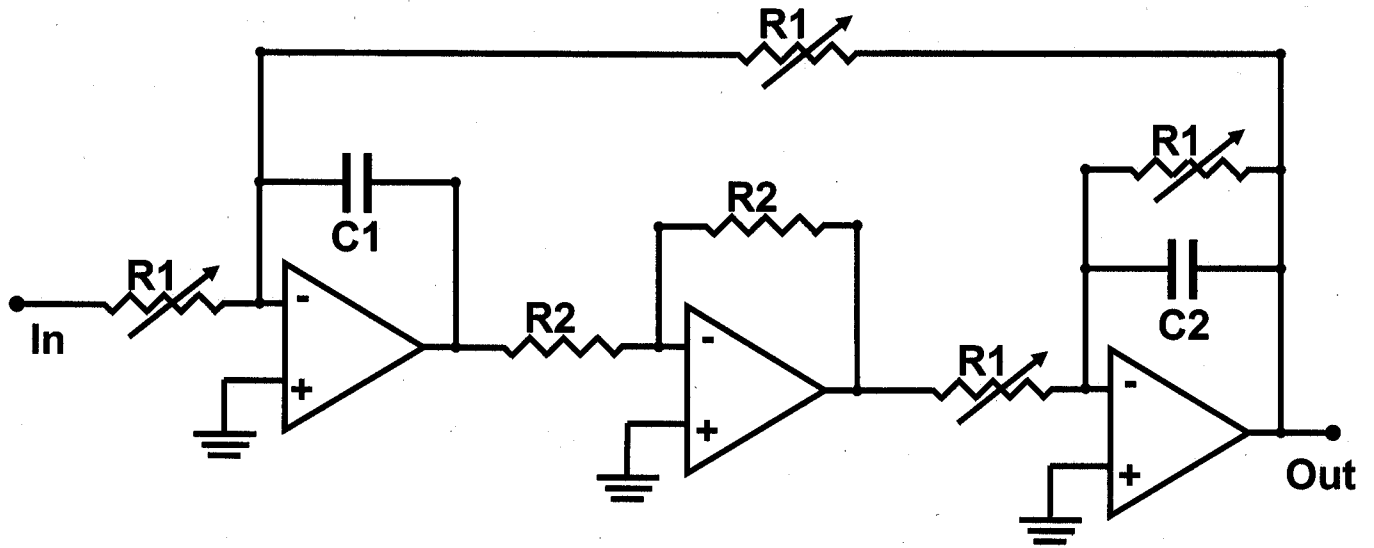


Fig. 4

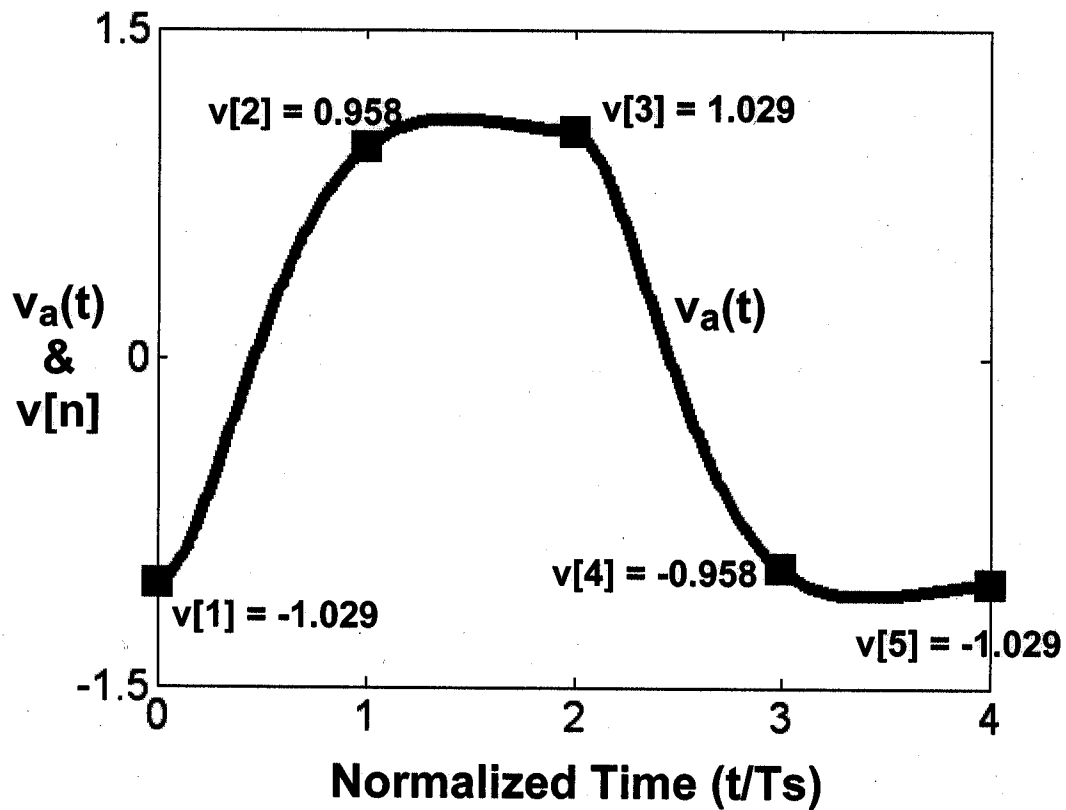


Fig. 5

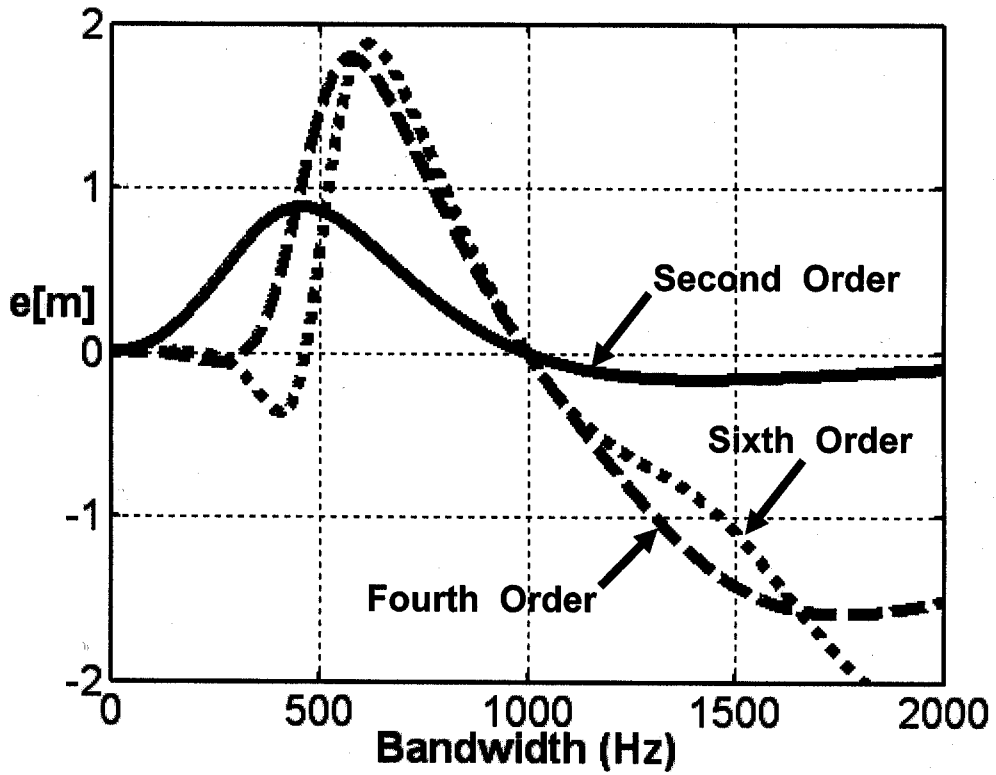


Fig. 6

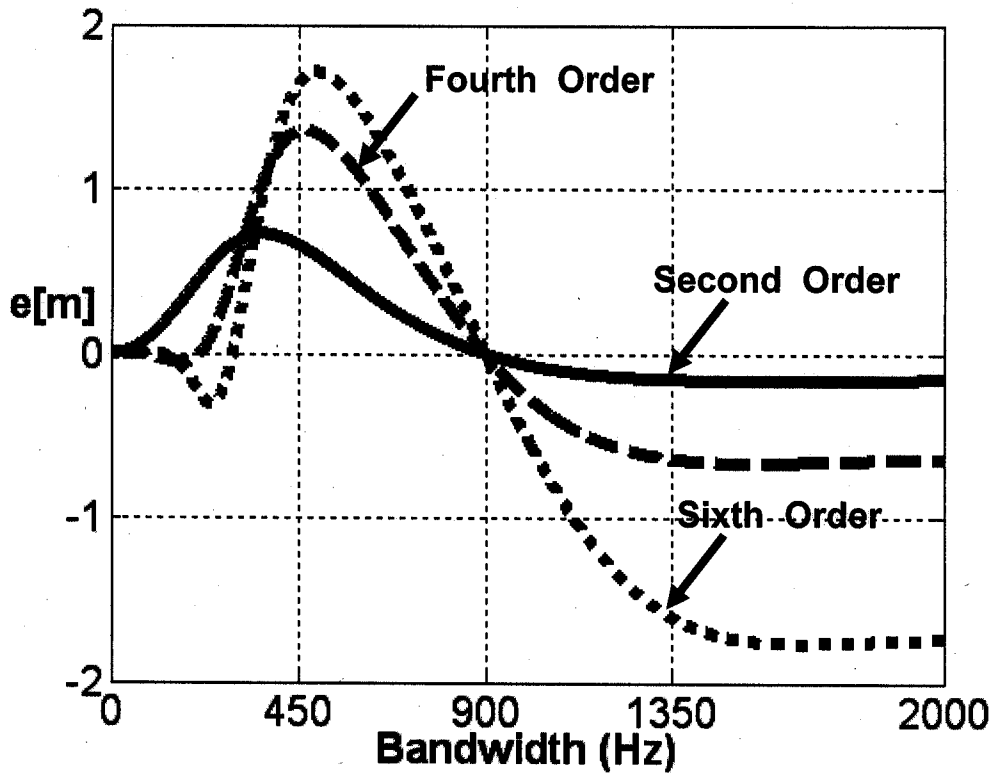


Fig. 7

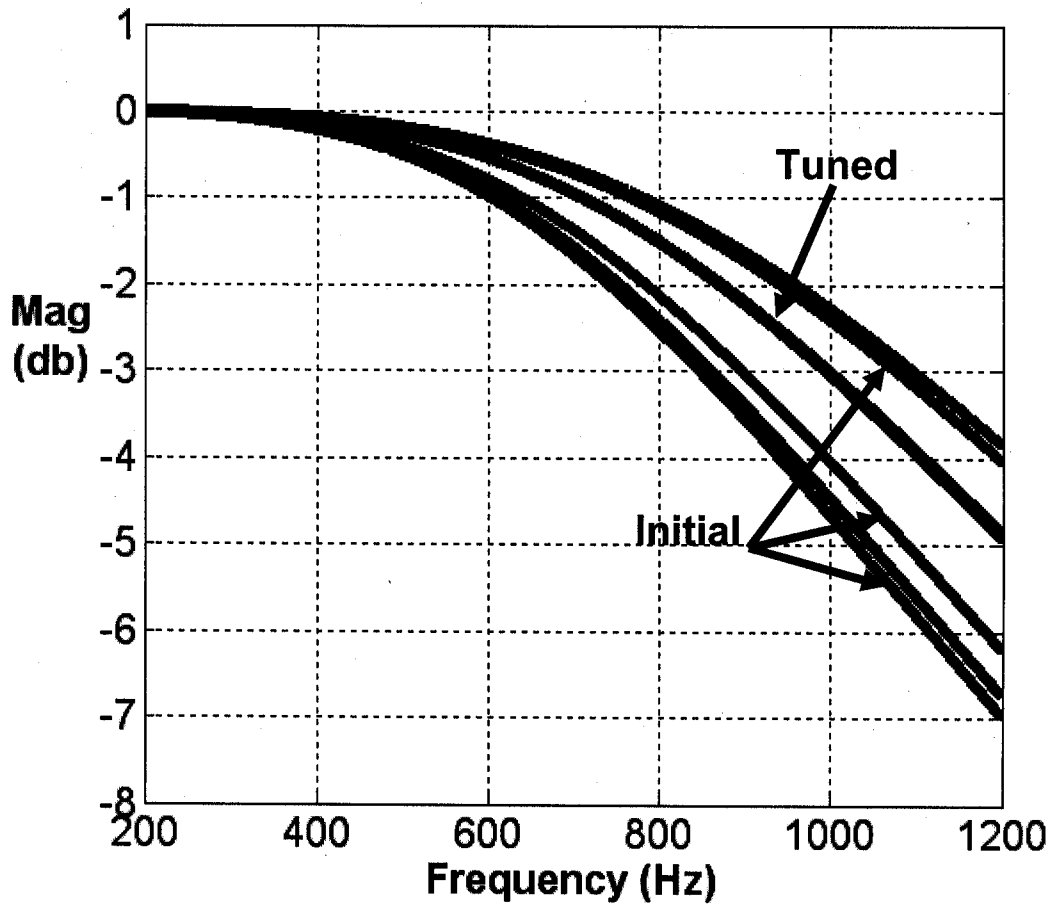


Fig. 8