2015 American Control Conference
Palmer House Hilton
July 1-3, 2015. Chicago, IL, USA

# The Multinotch, Part II: Extra Precision Via $\Delta$ Coefficients

Daniel Y. Abramovitch*

***Abstract*— In [1], we presented a new digital filter architecture, the multinotch, which minimized the computational latency while preserving numerical accuracy even in the presence of severe quantization. While this method is far more accurate than discretizing polynomial filters, it can still be susceptible to problems caused by a sample rate which is significantly higher than the frequencies of the features that the filter is trying to implement. This paper presents a modification, called $\Delta$ coefficients, which preserve all the positive properties of the multinotch while dramatically increasing the numerical accuracy over a large frequency range.**

## I. Introduction

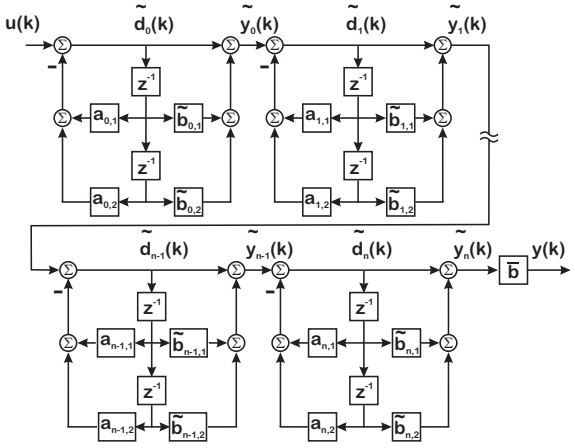


Fig. 1. The multinotch filter structure, presented in [1].

The multinotch [1], [2] shown in Figure 1 presents a new digital filter architecture, which minimizes the computational latency while preserving numerical accuracy even in the presence of severe quantization. This is especially useful for minimal latency control of mechatronic systems. The defining equations for the multinotch filter in [1] are given by

$$\tilde{d}_0(k) = D_{P,0}(k-1) + u(k), \tag{1}$$

$$\tilde{d}_i(k) = \sum_{j=0}^{i} D_{P,j}(k-1) + \sum_{j=0}^{i-1} F_{P,j}(k-1) + u(k), \tag{2}$$

for $i \geq 1$, and

$$\tilde{y}_i(k) = \sum_{j=0}^{i} D_{P,j}(k-1) + \sum_{j=0}^{i} F_{P,j}(k-1) + u(k), \tag{3}$$

for $i \geq 0$, where the delay precalculations are

$$D_{P,j}(k-1) = -a_{j,1}\tilde{d}_j(k-1) - a_{j,2}\tilde{d}_j(k-2) \tag{4}$$

*Daniel Y. Abramovitch is a system architect in the Mass Spectrometry Division, Agilent Technologies, 5301 Stevens Creek Blvd., M/S: 3U-DG, Santa Clara, CA 95051 USA, danny@agilent.com

and the output precalculations are

$$F_{P,j}(k-1) = \tilde{b}_{j,1}\tilde{d}_j(k-1) + \tilde{b}_{j,2}\tilde{d}_j(k-2). \tag{5}$$

Note that as the signal moves through biquad stages, sums get added to the precalculation. However, because these are all scaled the same, the sums that have already been computed can be reused. The final output, $y(k)$ is obtained simply from the last biquad output, $\tilde{y}_n$, as

$$y(k) = \bar{b}\tilde{y}_n(k), \text{ where } \bar{b} = b_{n,0}b_{n-1,0}\cdots b_{1,0}b_{0,0}. \tag{6}$$

This structure, shown in Figure 1 has several advantages [1], but a few key ones are:

- It retains the biquad form with the same poles and zeros as the original filter. The coefficients for the individual biquad sections can be computed independently of each other, retaining most of the physical intuition in the filter, even after discretization.
- $\tilde{d}_i(k)$ and $\tilde{y}_i(k)$ are computed from summing precalculated terms with the current input, $u(k)$.
- Once $u(k)$ is available, the computation of $y(k)$ involves two additions of precalculated terms and one multiplication by $\bar{b}$. This means that the computational latency between the measurement input and the filter output is very small and independent of filter length.

While this method is far more accurate than discretizing polynomial filters, it can still be susceptible to problems caused by a sample rate which is significantly higher than the the frequencies of the features that the filter is trying to implement. This paper presents a modification, called $\Delta$ coefficients, which preserve all the positive properties of the multinotch while increasing the numerical accuracy over a large frequency range. While the $\Delta$ coefficients draw inspiration from the $\delta$ transformation [3], [4], [5], [6], only the coefficients and not the signal space are modified.

## II. Multinotch Filter Coefficients

| | |
|---|---|
| $f_{N,i}$ | Center frequency of numerator (Hz) |
| $\omega_{N,i}$ | Center frequency of numerator (rad/s) |
| $Q_{N,i}$ | Quality factor of numerator |
| $\zeta_{N,i} = \frac{1}{Q_{N,i}}$ | Damping factor of numerator |
| $f_{D,i}$ | Center frequency of denominator (Hz) |
| $\omega_{D,i}$ | Center frequency of denominator (rad/s) |
| $Q_{D,i}$ | Quality factor of denominator |
| $\zeta_{D,i} = \frac{1}{Q_{D,i}}$ | Damping factor of denominator |

TABLE I

Physical coefficients used to specify a biquad section.

In [1] the filter was designed using the analog specification parameters of Table I and then digitized using pole-zero matching [7]. The biquad form means that there are no excess zeros to consider. The direct feedthrough for each digital biquad, $b_{i,0}$, was factored out, to be used in the computation of $\bar{b}$. It can be used as is or can be altered so that, for example, the DC gain of the biquad section will be 1.

The individual biquad coefficients are calculated as follows. For $a_{i,2}$, $\tilde{b}_{i,2}$, and $T_S = \frac{1}{f_S}$ we have

$$a_{i,2} = e^{-2\omega_{D,i}T_S\zeta_{D,i}} \text{ and } \tilde{b}_{i,2} = e^{-2\omega_{N,i}T_S\zeta_{N,i}}. \quad (7)$$

Whether the poles (or zeros) are a complex pair depends upon $|\zeta_{D,i}|$ ($|\zeta_{N,i}|$). For $|\zeta_{D,i}| < 1$ we have a complex pair of poles and so

$$a_{i,1} = -2e^{-\omega_{D,i}T_S\zeta_{D,i}} \cos\left(\omega_{D,i}T_S\sqrt{1-\zeta_{D,i}^2}\right). \quad (8)$$

If $|\zeta_{N,i}| < 1$ we have a complex pair of zeros and so

$$\tilde{b}_{i,1} = -2e^{-\omega_{N,i}T_S\zeta_{N,i}} \cos\left(\omega_{N,i}T_S\sqrt{1-\zeta_{N,i}^2}\right). \quad (9)$$

While these two cases represent cases when the desired filters have very sharp peaks or notches (for example to equalize a response with very sharp notches or peaks), there are other possibilities. For example setting $|\zeta_{D,i}| = 1$ ($|\zeta_{N,i}| = 1$) means that the poles (zeros) are real and equal, so

$$a_{i,1} = -2e^{-\omega_{D,i}T_S\zeta_{D,i}} \text{ and } \tilde{b}_{i,1} = -2e^{-\omega_{N,i}T_S\zeta_{N,i}}. \quad (10)$$

Finally, if $|\zeta_{D,i}| > 1$ ($|\zeta_{N,i}| > 1$) means that the poles (zeros) are real and distinct, so $a_{i,1}$ ($\tilde{b}_{i,1}$ are given by using the cosh relation:

$$a_{i,1} = -2e^{-\omega_{D,i}T_S\zeta_{D,i}} \cosh\left(\omega_{D,i}T_S\sqrt{\zeta_{D,i}^2-1}\right) \quad (11)$$

and

$$\tilde{b}_{i,1} = -2e^{-\omega_{N,i}T_S\zeta_{D,i}} \cosh\left(\omega_{N,i}T_S\sqrt{\zeta_{N,i}^2-1}\right). \quad (12)$$

The entire conversion routine, which turns the physical parameters of Table I into discrete filter coefficients can be implemented in a short Matlab or Octave function.

## III. Effects of a Relatively Small $T_S$

A significant problem can arise when $\omega_{D,i}T_S$ or $\omega_{N,i}T_S$ get very small. Equation 7 implies

$$\lim_{\omega_{D,i}T_S\to 0} a_{i,2} = 1 \quad \text{and} \quad \lim_{\omega_{N,i}T_S\to 0} \tilde{b}_{i,2} = 1. \quad (13)$$

Similarly, since $\cos(0) = \cosh(0) = 1$, we can see from Equations 8, 10, and 11 (and 9, 10, and 12) that

$$\lim_{\omega_{D,i}T_S\to 0} a_{i,1} = -2, \quad \text{and} \quad \lim_{\omega_{N,i}T_S\to 0} \tilde{b}_{i,1} = -2. \quad (14)$$

This means that in the limit, both the numerator and denominator approach $P(z) = z^2 - 2z + 1$, which has two roots at $z = 1$. The effect of increased sample rate relative to a given feature frequency is to push that feature closer and closer to $z = 1$ on the $z$ plane. While it is difficult to see on a z-plane plot, Tables II and III show two examples of the effect of sample rate on the quantized coefficients, by translating those back into poles and zeros. The details of how the quantized values are obtained are in Section V, but for now we see that Tables II and III demonstrate that features in continuous time wind up as extremely small perturbations around $z = 1$ in discrete time. We can see from both tables that for the lower sample rate, the poles and zeros are relatively consistent despite quantization. As the sample rate goes higher, the poles and zeros corresponding to quantized values vary slightly, but these result in significant performance differences. Only the $\Delta$ coefficients, to be introduced in Section IV, result in poles and zeros near the ones from the unquantized filters.

We will see in Section V that these minor variations due to fixed point math give significant performance issues, as demonstrated by generating frequency responses of the different filters.

## IV. $\Delta$ Coefficients

Fixing this problem in fixed point math might be attempted with adding extra bits to the fractional portion. We will see in the examples of Section V that this has limited positive effect, unless the number of added bits gets significant. If we add enough bits to achieve numerical accuracy, we risk making our multiplicands too wide to fit into the hardware multiplier blocks of the desired real-time processor or FPGA. For example, Xilinx FPGA multiplier blocks in Version 7, all feature multiplier blocks that multiply a pair of twos-compliment numbers that are 25 bits by 18 bits [8]. The point is that in FPGAs and fixed point DSPs, it is necessary to implement multiply and accumulate (MAC) operations with relatively narrow fixed point numbers. It is certainly possible to extend the width of multiplies using extra multiplier blocks in an FPGA or extra code in a DSP, but at the cost of extra delay [9].

Another option is to convert all the signals in the system away from a $z$ Transform and to a $\delta$ operator form [3], [4], [5], [6]. In [10], the filter is also broken into biquad sections and these are transformed using the $\delta$ operator. However, this method adds some complexity to the filter operation. The method proposed below simply restores accuracy to the discrete coefficients without changing the basic math operations. Some further comparison will be done in Section VI. The method here borrows the basic idea from $\delta$ operator methods but notes that we are not concerning ourselves with the $z$ terms per se being clustered around $z = 1$, but the roots of the biquad polynomials. As we saw in Section III, the $a_{i,1}$ and $\tilde{b}_{i,1}$ terms go to $-2$ while the $a_{i,2}$ and $\tilde{b}_{i,2}$ terms go to 1. With this understanding, we can define:

$$a_{i,1} = -2 + a_{i,1\Delta} \quad \text{so} \quad a_{i,1\Delta} = a_{i,1} + 2, \quad (15)$$

$$a_{i,2} = 1 + a_{i,2\Delta} \quad \text{so} \quad a_{i,2\Delta} = a_{i,1} - 1, \quad (16)$$

$$\tilde{b}_{i,1} = -2 + \tilde{b}_{i,1\Delta} \quad \text{so} \quad \tilde{b}_{i,1\Delta} = \tilde{b}_{i,1} + 2, \text{ and} \quad (17)$$

$$\tilde{b}_{i,2} = 1 + \tilde{b}_{i,2\Delta} \quad \text{so} \quad \tilde{b}_{i,2\Delta} = \tilde{b}_{i,1} - 1. \quad (18)$$

Now, $a_{i,1\Delta}$, $a_{i,2\Delta}$, $\tilde{b}_{i,1\Delta}$, and $\tilde{b}_{i,2\Delta}$ are small numbers. The smaller $\omega_{D,i}T_S$ gets, the smaller $a_{i,1\Delta}$ and $a_{i,2\Delta}$ get.

| $f_S$ | Quantization | $z_{1,2}$ | $p_{1,2}$ | $k$ |
|---|---|---|---|---|
| 10000 | **Float** | 9.972435e-001 ± 6.273633e-002 | 9.902495e-001 ± 6.181130e-002 | 1 |
| 10000 | **s2.16** | 9.972381e-001 ± 6.280493e-002 | 9.902495e-001 ± 6.173522e-002 | 1 |
| 10000 | **s2.23** | 9.972435e-001 ± 6.273632e-002 | 9.902495e-001 ± 6.181097e-002 | 1 |
| 10000 | **Δ s2.16** | 9.972436e-001 ± 6.273490e-002 | 9.902496e-001 ± 6.181050e-002 | 1 |
| $f_S$ | Quantization | $z_{1,2}$ | $p_{1,2}$ | $k$ |
| 100000 | **Float** | 9.999017e-001 ± 6.282160e-003 | 9.991955e-001 ± 6.228970e-003 | 1 |
| 100000 | **s2.16** | 9.999008e-001 ± 5.523423e-003 | 9.991913e-001 ± 6.717367e-003 | 1 |
| 100000 | **s2.23** | 9.999017e-001 ± 6.280814e-003 | 9.991955e-001 ± 6.229847e-003 | 1 |
| 100000 | **Δ s2.16** | 9.999017e-001 ± 6.282048e-003 | 9.991955e-001 ± 6.228704e-003 | 1 |
| $f_S$ | Quantization | $z_{1,2}$ | $p_{1,2}$ | $k$ |
| 1000000 | **Float** | 9.999919e-001 ± 6.282645e-004 | 9.999213e-001 ± 6.233415e-004 | 1 |
| 1000000 | **s2.16** | 0.999999999999995, 0.999969481956212 | 0.999999999999971, 0.999832150759166 | 1 |
| 1000000 | **s2.23** | 9.999919e-001 ± 6.904864e-004 | 9.999213e-001 ± 5.928139e-004 | 1 |
| 1000000 | **Δ s2.16** | 9.999919e-001 ± 6.283486e-004 | 9.999213e-001 ± 6.234487e-004 | 1 |

TABLE II

FILTER POLES AND ZEROS UNDER QUANTIZATION AT DIFFERENT SAMPLE FREQUENCIES. LEAD DESIGN PARAMETERS: $f_N = 100$, $Q_N = 40$, $f_D = 100$, $Q_D = 4$.

| $f_S$ | Quantization | $z_{1,2}$ | $p_{1,2}$ | $k$ |
|---|---|---|---|---|
| 10000 | **Float** | 9.676381e-001 ± 5.270507e-002 | 9.335457e-001 ± 1.019989e-001 | 1 |
| 10000 | **s2.16** | 9.676356e-001 ± 5.274999e-002 | 9.335393e-001 ± 1.020527e-001 | 1 |
| 10000 | **s2.23** | 9.676381e-001 ± 5.270573e-002 | 9.335457e-001 ± 1.019985e-001 | 1 |
| 10000 | **Δ s2.16** | 9.676385e-001 ± 5.271557e-002 | 9.335460e-001 ± 1.020009e-001 | 1 |
| $f_S$ | Quantization | $z_{1,2}$ | $p_{1,2}$ | $k$ |
| 100000 | **Float** | 9.968486e-001 ± 5.424303e-003 | 9.936777e-001 ± 1.081442e-002 | 1 |
| 100000 | **s2.16** | 9.968414e-001 ± 5.983327e-003 | 9.936751e-001 ± 1.061067e-002 | 1 |
| 100000 | **s2.23** | 9.968485e-001 ± 5.433827e-003 | 9.936776e-001 ± 1.081786e-002 | 1 |
| 100000 | **Δ s2.16** | 9.968486e-001 ± 5.422943e-003 | 9.936777e-001 ± 1.081251e-002 | 1 |
| $f_S$ | Quantization | $z_{1,2}$ | $p_{1,2}$ | k |
| 1000000 | **Float** | 9.996857e-001 ± 5.439689e-004 | 9.993713e-001 ± 1.087596e-003 | 1 |
| 1000000 | **s2.16** | 1.000000000000061, 0.999359121080277 | 1.000000000000042, 0.998733501182532 | 1 |
| 1000000 | **s2.23** | 9.996857e-001 ± 5.087695e-004 | 9.993712e-001 ± 1.128530e-003 | 1 |
| 1000000 | **Δ s2.16** | 9.996857e-001 ± 5.509819e-004 | 9.993713e-001 ± 1.088241e-003 | 1 |

TABLE III

FILTER POLES AND ZEROS UNDER QUANTIZATION AT DIFFERENT SAMPLE FREQUENCIES. LEAD DESIGN PARAMETERS: $f_N = 100$, $Q_N = 1$, $f_D = 200$, $Q_D = 1$.

Likewise the smaller $\omega_{N,i}T_S$ gets, the smaller $\tilde{b}_{i,1\Delta}$ and $\tilde{b}_{i,2\Delta}$ get. However, we can split up the signal multiplications

$$a_{i,1}d_i(k-1) = -2d_i(k-1) + a_{i,1\Delta}d_i(k-1), \quad (19)$$

$$\tilde{b}_{i,1}d_i(k-1) = -2d_i(k-1) + \tilde{b}_{i,1\Delta}d_i(k-1), \quad (20)$$

$$a_{i,2}d_i(k-2) = d_i(k-2) + a_{i,2\Delta}d_i(k-2), \quad (21)$$

and

$$\tilde{b}_{i,2}d_i(k-2) = d_i(k-2) + \tilde{b}_{i,2\Delta}d_i(k-2). \quad (22)$$

In each of these, the first multiplication on the right is either a trivial multiply by 2, accomplished with a shift to the left by one bit, or it is a more trivial multiply by 1, accomplished by doing nothing. We can now concentrate on making the second multiply more accurate. In real numbers,

$$a_{i,1\Delta}d_i(k-1) = (2^E a_{i,1\Delta})d_i(k-1)2^{-E}. \quad (23)$$

If we scale up $a_{i,1\Delta}$ by a number to maximize the number of significant digits in the fixed point representation, our multiplication will have the maximum accuracy. We can scale the product down by that same number for adding into the precalc sum. Likewise, we can do the same thing for the other $\Delta$ coefficients:

$$\tilde{b}_{i,1\Delta}d_i(k-1) = (2^E\tilde{b}_{i,1\Delta})d_i(k-1)2^{-E}, \quad (24)$$

$$a_{i,2\Delta}d_i(k-2) = (2^E a_{i,2\Delta})d_i(k-2)2^{-E}, \quad (25)$$

and

$$\tilde{b}_{i,2\Delta}d_i(k-2) = (2^E\tilde{b}_{i,2\Delta})d_i(k-2)2^{-E}. \quad (26)$$

### A. Computing Scaling

How do we compute the scaling factor, $2^{-E}$? Consider a coefficient, $c_i$:

$$\log_2|c_i| = x_i \text{ means } 2^{x_i} = |c_i|. \quad (27)$$

Let's say we want to do multiplies with a coefficient that has a magnitude between 1 and 2. In this case we want

$$1 \le 2^{-E_i}|c_i| < 2 \text{ or } 0 \le \log_2 2^{-E_i}|c_i| < 1 \quad (28)$$

$$\text{which means} \qquad 0 \le x_i - E_i < 1. \quad (29)$$

$E_i$ represents the integer part of $\log_2|c_i|$ so,

$$\text{floor}(\log_2|c_i|) \text{ will give us } E_i. \quad (30)$$

If we divide by $2^{E_i}$, it's like multiplying by $2^{-E_i}$. What is the effect of this? If $E_i$ is positive, then we are shrinking the magnitude of the coefficient by a power of 2. If $E_i$ is negative, then we are raising the magnitude of the coefficient by a power of 2.

Now, for each of the floating point versions of $a_{i,1\Delta}$, $a_{i,2\Delta}$, $\tilde{b}_{i,1\Delta}$, and $\tilde{b}_{i,2\Delta}$ we could have a separate value of $E_i$. However, experience has shown that if the multinotch poles and zeros are grouped so that biquads have poles and zeros that are as close as possible to each other, a single value of $2^{-E_i}$ can be used for each biquad. If we pick

$$E_i = \max(E_{i,a1}, E_{i,a2}, E_{i,b1}, E_{i,b2}), \tag{31}$$

that is we pick the maximum of the negative exponents for the $\Delta$ coefficients, then we will be multiplying by the minimum $2^{-E_i}$.

### B. Implementing $\Delta$ Coefficients

Now, rewriting (4)

$$\begin{aligned} D_{P,i}(k-1) &= -a_{i,1}\tilde{d}_i(k-1) - a_{i,2}\tilde{d}_i(k-2) && (32)\\ &= (2 - a_{i,1\Delta})\tilde{d}_i(k-1) \\ &\quad -(1 + a_{i,2\Delta})\tilde{d}_i(k-2) && (33)\\ &= D_{PW,i}((k-1) + D_{PF,i}((k-1) && (34)\end{aligned}$$

where

$$\begin{aligned} D_{PW,i}(k-1) &= 2\tilde{d}_i(k-1) - \tilde{d}_i(k-2) \text{ and} && (35)\\ D_{PF,i}(k-1) &= -a_{i,1\Delta}\tilde{d}_i(k-1) - a_{i,2\Delta}\tilde{d}_i(k-2) && (36)\end{aligned}$$

The fractional precalc can be done in two steps as:

$$\begin{aligned} D_{PFL,i}(k-1) &= -(2^{-E_i}a_{i,1\Delta})\tilde{d}_i(k-1) \\ &\quad -(2^{-E_i}a_{i,2\Delta})\tilde{d}_i(k-2). && (37)\\ D_{PF,i}(k-1) &= 2^{E_i}D_{PFL,i}(k-1). && (38)\end{aligned}$$

The coefficients in (37) are computed from the floating point numbers, $2^{-E_i}a_{i,1\Delta}$, and $2^{-E_i}a_{i,2\Delta}$, before being converted to fixed point. Once the multiplication has been done with high precision, the product is shifted back in (38) for addition with $D_{PW,i}(k-1)$. If the scaled down product is insignificant compared to $D_{PW,i}(k-1)$. However, the high precision multiplication means that if the product is significant, it is also accurate.

We repeat the process with the output precalcs. Rewriting (5)

$$\begin{aligned} F_{P,i}(k-1) &= \tilde{b}_{i,1}\tilde{d}_i(k-1) + \tilde{b}_{i,2}\tilde{d}_i(k-2) && (39)\\ &= (2 - \tilde{b}_{i,1\Delta})\tilde{d}_i(k-1) \\ &\quad -(1 + \tilde{b}_{i,2\Delta})\tilde{d}_i(k-2) && (40)\\ &= F_{PW,i}((k-1) + F_{PF,i}((k-1) && (41)\end{aligned}$$

where

$$\begin{aligned} F_{PW,i}(k-1) &= 2\tilde{d}_i(k-1) - \tilde{d}_i(k-2) \text{ and} && (42)\\ F_{PF,i}(k-1) &= -\tilde{b}_{i,1\Delta}\tilde{d}_i(k-1) - \tilde{b}_{i,2\Delta}\tilde{d}_i(k-2) && (43)\end{aligned}$$

The fractional precalc can be done in two steps as:

$$\begin{aligned} F_{PFL,i}(k-1) &= -(2^{-E_i}\tilde{b}_{i,1\Delta})\tilde{d}_i(k-1) \\ &\quad -(2^{-E_i}\tilde{b}_{i,2\Delta})\tilde{d}_i(k-2). && (44)\\ F_{PF,i}(k-1) &= 2^{E_i}F_{PFL,i}(k-1). && (45)\end{aligned}$$

The equations above illustrate one of the beauties of the $\Delta$ coefficient approach. While there are a few extra additions and right shifts of multiplied values in the precalculation portion of the filter, there are no extra multiplies. Instead many of the existing multiplies have been made far more accurate. Additions and shifts are extremely easy operations in digital hardware, and therefore the added computational burden of this extra accuracy in very small.

## V. Examples

| **Example 1** | | | | |
|---|---|---|---|---|
| **Biquad #** | $f_{N,n}$ **(Hz)** | $Q_n$ | $f_{N,d}$ **(Hz)** | $Q_d$ |
| 1 | 100 | 40 | 100 | 4 |
| **Example 2** | | | | |
| **Biquad #** | $f_{N,n}$ **(Hz)** | $Q_n$ | $f_{N,d}$ **(Hz)** | $Q_d$ |
| 1 | 100 | 1 | 200 | 1 |

TABLE IV
FILTER PARAMETERS FOR BOTH EXAMPLES.

In order to compare filters, a set of filter parameters was chosen in the form of sets analog biquad parameters, such as those in Table IV. Unlike the examples in [1], only a single biquad was needed to demonstrate the desired effects. For a given set of biquad parameters, the sample frequency was varied between 10 kHz, 100 kHz, and 1 MHz. In the first example from Table IV, a high Q notch filter was chosen, centered at 100 Hz. In the second example, a lead-lag filter was implemented where the lead natural frequency was set to 100 Hz and the lag natural frequency was set to 200 Hz. The poles and zeros from these examples have already been presented in Tables II and III Section III.

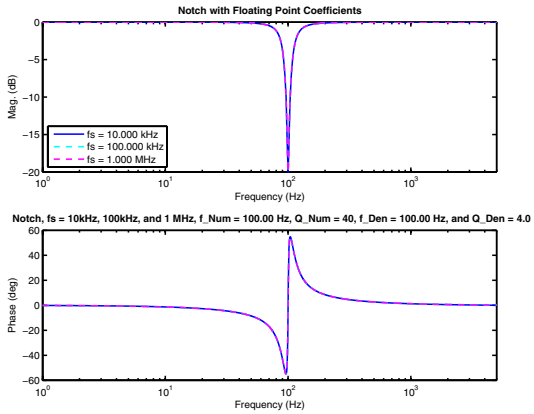


Fig. 2. Notch with $f_n$ and $f_d$ at 100 Hz, $Q_n = 40$, $Q_d = 4$, no quantization. Sample frequencies are $f_S = 10$kHz, 100kHz, and 1MHz. With no quantization, there is effectively no difference.

The filter design parameters are specified by Table IV. These parameters were then translated into digital filter in

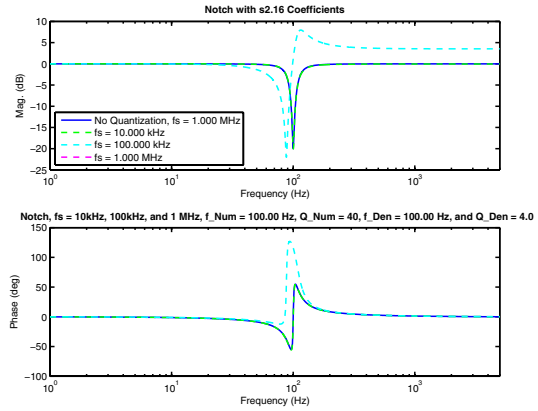


Fig. 3. Notch with $f_n$ and $f_d$ at 100 Hz, $Q_n = 40$, $Q_d = 4$, quantized to s2.16. Sample frequencies are $f_S = 10$kHz, 100kHz, and 1MHz.

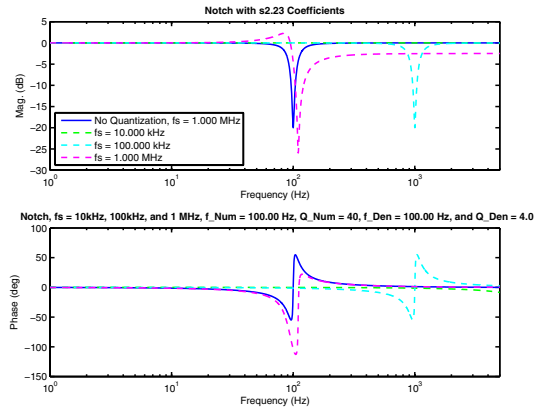


Fig. 4. Notch with $f_n$ and $f_d$ at 100 Hz, $Q_n = 40$, $Q_d = 4$, quantized to s2.23. Sample frequencies are $f_S = 10$kHz, 100kHz, and 1MHz.

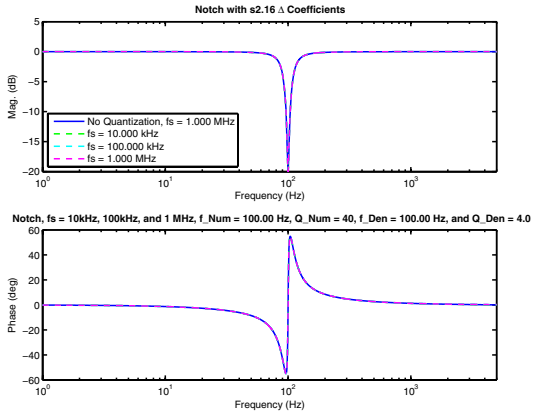


Fig. 5. Notch with $f_n$ and $f_d$ at 100 Hz, $Q_n = 40$, $Q_d = 4$, using $\Delta$ coefficients, quantized to s2.16. Sample frequencies are $f_S = 10$kHz, 100kHz, and 1MHz. The $\Delta$ coefficients restore numerical accuracy.

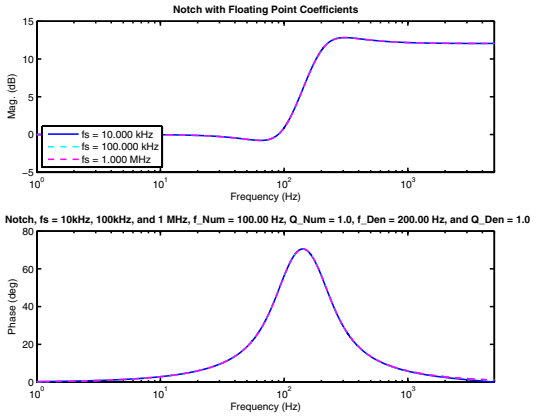


Fig. 6. Lead with $f_n$ at 100 Hz and $f_d$ at 200 Hz, $Q_n = 1$, $Q_d = 1$, no quantization. Sample frequencies are $f_S = 10$kHz, 100kHz, and 1MHz. With no quantization, there is effectively no difference.

the form of a single biquad, essentially a single notch in the same form as the filters in [1]. The coefficients of the digital filter were then scaled up by a quantization factor, say $2^{16} - 1$ for an s2.16 quantization. The scaled up coefficients were then fixed (fractional portion removed) and scaled down by the same quantization factor. Thus, floating point numbers were made to represent fixed point coefficients. Frequency responses were computed for the filters with floating point coefficients and with quantized coefficients. The same original filter specifications were used to show the variation with sample frequency. Finally, these were compared to a quantized biquad implemented with $\Delta$ coefficients. The three sample rates of $f_S = 10$ kHz, 100 kHz, and 1 MHz were adequate to demonstrate the result.

With floating point coefficients, we can see from Figures 2 and 6 that the filters are essentially unaffected by the change in sample frequency. Quantizing the filter coefficients to a s2.16 format, as shown in Figures 3 and 7 shows that while the quantized filter is accurate at $f_S = 10$ kHz, it becomes inaccurate at $f_S = 100$ kHz. With $f_S = 1$ MHz, the plot is off scale. The situation improves some by adding more bits to a s2.23 format, as shown in Figures 4 and 8, however none of these are at all accurate. In fact, both figures demonstrate one of the dangers of quantizing such parameter values, the nonlinear degradation. In both examples the filter response at $f_S = 1$ MHz is more accurate than that at $f_S = 100$ kHz. The success of the $\Delta$ coefficients is demonstrated in Figures 5 and 9. Using only s2.16 $\Delta$ parameterized coefficients, we have essentially restored the filter accuracy for all three sample frequencies.

## VI. $\Delta$ Coefficients Versus $\delta$ Parameterization and Floating Point

In contrast with the $\Delta$ coefficients which maintain the delay form ($z^{-1}$) of the discrete filter while improving the coefficient accuracy, the $\delta$ parameterization [3], [4], [5], [6] pushes the discrete parameters closer to the the continuous time parameters and the filter calculation are transformed from step form to a differential form. Most similar to the method here is [10] which breaks the filter into biquads as well and then applies the $\delta$ parameterization to each biquads. The assumption when using this is that the filter has been somehow discretized already and then is reparameterized using a forward rectangular rule integration approximation. In the $\Delta$ coefficient formulation, each biquad is discretized individually and the matched pole-zero method provides excellent agreement to continuous frequency responses for

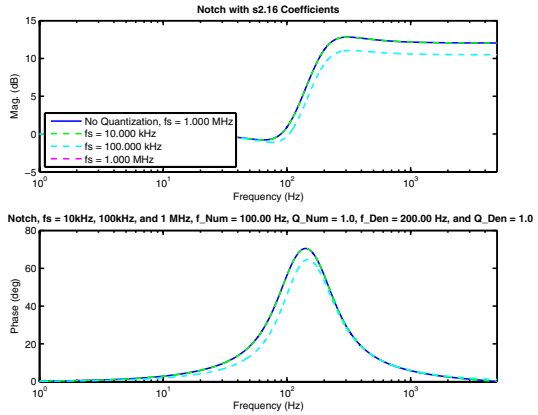


Fig. 7. Lead with $f_n$ at 100 Hz and $f_d$ at 200 Hz, $Q_n = 1$, $Q_d = 1$, quantized to s2.16. $f_S =$ 10kHz, 100kHz, and 1MHz.

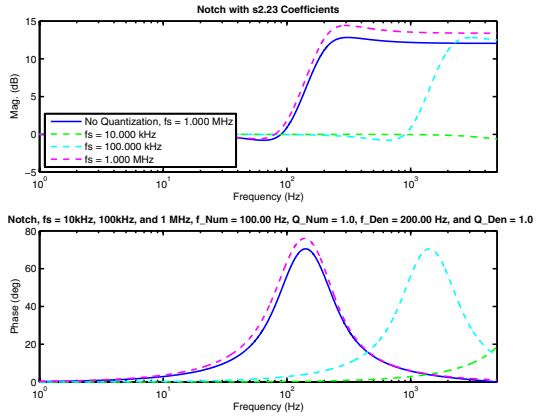


Fig. 8. Lead with $f_n$ at 100 Hz and $f_d$ at 200 Hz, $Q_n = 1$, $Q_d = 1$, quantized to s2.23. $f_S =$ 10kHz, 100kHz, and 1MHz.

biquads. Further comparisons will be made in [11]. The $\Delta$ coefficients approximate floating point coefficients in a computationally inexpensive way. The filter computations are done using the delay form, which is slightly less complicated than the differential form of the $\delta^{-1}$ block of [10]. Finally, the $\delta$ parameterization is useful primarily when the sample rate is high relative to the dynamics being filtered, when $\omega_0 T_S = 2\pi f_0 / f_S$ gets very small. The forward rectangular rule becomes much more inaccurate for slow sample rates. Mechatronic systems are known for having a wide spread in dynamics, and so the $\delta$ parameterization could have some issues not present in the $\Delta$ coefficients [11].

In minimal latency control of a mechatronic system, the original inspiration for the multinotch [1], floating point computations may take too long. A native mode 25 bit $\times$ 18 bit floating point multiply and add in a Xilinx DSP48E will have a latency of 4 clock cycles [8]. Single precision floating point multiplies take 8 clock cycles, but the additions must be separately and take 11 clock cycles [9].

## VII. Conclusions

The multinotch represents a new way of structuring digital filters so as combine physical intuition, numerical stability, and low, fixed latency into a single architecture [1]. However, even the multinotch can degrade severely when the sample

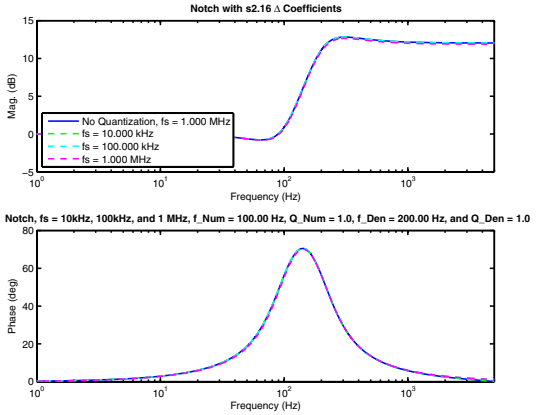


Fig. 9. Lead with $f_n$ at 100 Hz and $f_d$ at 200 Hz, $Q_n = 1$, $Q_d = 1$, using $\Delta$ coefficients, quantized to s2.16. $f_S =$ 10kHz, 100kHz, and 1MHz. The $\Delta$ coefficients have restored the numerical accuracy.

frequency is several orders of magnitude higher than the frequencies of filter features. In order to correct this while still maintaining fixed point math, suitable for high speed, real time implementation on an FPGA or a DSP, the $\Delta$ coefficient parameterization has been presented. This restores the performance of the multinotch, while adding only a few extra precalculations and no extra bits. The $\Delta$ coefficients extend the range and accuracy of the multinotch without affecting the physical intuition of the filter, which we have asserted is highly desirable in debugging the physical behavior of a system.

## References


[1] D. Y. Abramovitch, "The Multinotch, Part I: A low latency, high numerical fidelity filter for mechatronic control systems," in *Submitted to the 2015 American Control Conference*, (Chicago, IL), AACC, IEEE, July 2015.

[2] D. Y. Abramovitch and C. R. Moon, "Cascaded digital filters with reduced latency," International Application Published Under the Patent Cooperation Treaty WO 2012/118483, World Intellectual Property Organization, September 9 2012.

[3] R. H. Middleton and G. C. Goodwin, "Improved finite word length characteristics in digital control using delta operators," *IEEE Transactions on Automatic Control*, vol. 31, pp. 1015–1021, November 1986.

[4] R. M. Goodall and B. J. Donoghue, "Very high sample rate digital filters using the $\delta$ operator," *IEE Proceedings-G*, vol. 140, pp. 199–206, June 1993.

[5] G. Li and M. Gevers, "Comparative study of finite wordlength effects in shift and delta operator parameterizations," *IEEE Transactions on Automatic Control*, vol. 38, pp. 803–807, May 1993.

[6] G. C. Goodwin, J. I. Yuz, J. C. Agüero, and M. Cea, "Sampling and sampled-data models," in *Proceedings of the 2010 American Control Conference*, (Baltimore, MD), AACC, IEEE, June 2010.

[7] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital Control of Dynamic Systems*. Menlo Park, California: Addison Wesley Longman, third ed., 1998.

[8] Xilinx, *7 Series DSP48E1 Slice Users Guide*, ug479 (v1.6) ed., August 7 2013.

[9] Xilinx, *LogiCORE IP Floating-Point Operator v6.0*, ds816 (v1.2) ed., January 18 2012.

[10] J. Kauraniemi, T. I. Laakso, I. Hartimo, and S. J. Ovaska, "Delta operator realizations of direct-form iir filters," *IEEE Transactions on Circuits and Systems–II: Analog and Digital Signal Processing*, vol. 45, pp. 41–52, January 1998.

[11] D. Y. Abramovitch, "A comparison of $\Delta$ coefficients and the $\delta$ parameterization," in *To be submitted to the 2016 American Control Conference*, (Boston, MA), AACC, IEEE, July 2016.