PROGRAMMABLE MULTIPLEXED SWITCHED-CAPACITOR FILTERS

X.F. Wania, D.A. Johns, and A.S. Sedra

Department of Electrical Engineering
University of Toronto
Toronto, Ontario
M5S 1A4, Canada

Abstract

This paper proposes an area efficient method to realize fully programmable SC filters through the use of time-sharing. Previously, fully programmable SC filters have been restricted to only second-order due to the large area requirement of the programmable capacitor arrays (PCAs). The proposed approach uses only 3 PCAs to implement an N'th order filter capable of having arbitrary transfer-functions.

1 Introduction

Switched-capacitor (SC) filters are fully-integrated, high-quality, frequency selective devices. These filters are used extensively in telecommunications, test instruments, and in other applications in the audio band. Their precision, compactness and low cost make them preferable to discrete, hybrid and digital filters in many applications [1].

With switched-capacitor (SC) filters, multiplexing (or time-sharing) has traditionally been used to reduce the number of op-amps [2-6]. In this paper, however, we choose to multiplex the input-summing capacitors of the integrators resulting in an area efficient method to realize N'th order fully programmable SC filters. The final design requires 3 programmable capacitor arrays (PCAs) 1 together with 2N clock phases. It should be mentioned here that without the use of multiplexing, the minimum number of PCAs required would be 2N; however, it is typical to use at least 3N PCAs to maintain reasonable filter performance with respect to sensitivity and dynamic range. This requirement for a large number of PCAs is the major reason that high-order fully programmable SC filters are not readily available. Of course, one drawback of multiplexing is the large number of clock phases required resulting in a shorter time available for charging and discharging capacitors.

To realize arbitrary transfer-functions, the common approach is to make use of a design based on cascade of biquads. The difficulty with this approach is that the optimum biquad structure depends strongly on the transfer-function being realized, (such as bandpass, high-pass notch, etc.) and on the value of the pole-Q. Also, pole-zero pairing, cascade ordering, and dynamic range scaling in designing a cascade of biquads would be tedious and time consuming. In the continuous-time domain, an alternate approach is to make use of the orthonormal ladder filter structure where a single structure is used for arbitrary transfer-functions and the filter is inherently scaled for optimum dynamic range [7]. The filter structure used in this paper is derived from an exact design technique based on the orthonormal ladder filter structure. The resulting filters are parasitic insensitive and have no delay-free loops – a necessary requirement to make use of multiplexing. As well, the same filters appear to have sensitivity and dynamic range performances comparable to an optimum design of a cascade of biquads. We call these filters, quasi-orthonormal SC state-space filters, since the orthonormal property is almost maintained in the discrete-time domain.

2 Design Procedure

In this section the filter structure and the multiplexing scheme used for implementing the programmable multiplexed SC filter is described.

2.1 Filter structure

A set of state-space equations is given by,

\[ \gamma \dot{\mathbf{X}}_n(\tau) = \mathbf{A}_n \mathbf{X}_n(\tau) + \mathbf{b}_n u_n(\tau) \] (1)

\[ \gamma \mathbf{Y}_n(\tau) = \mathbf{c}_n^T \mathbf{X}_n(\tau) + \mathbf{d}_n u_n(\tau) \] (2)

For an orthonormal ladder structure the state-space matrices take the form,

\[ \mathbf{A}_n = \begin{bmatrix} 0 & \alpha_1 & 0 & \cdots & 0 \\ -\alpha_1 & 0 & \alpha_2 & \cdots & 0 \\ 0 & -\alpha_2 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -\alpha_{N-1} & \alpha_N \end{bmatrix}, \quad \mathbf{b}_n = \begin{bmatrix} 0 \\ \vdots \\ -\alpha_{N-1} \\ \cdots \\ 0 \end{bmatrix} \] (3)

\[ \mathbf{c}_n^T = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_N \end{bmatrix}, \quad \mathbf{d}_n = d_n. \] (4)

To obtain an LDI-based realization, we use the transformation,

\[ \gamma = \frac{z - 1}{z^2}. \] (5)

Substituting for \( \gamma \) in Equation (1) and scaling the appropriate states, we obtain [8],

\[ (z - 1) \mathbf{X}(z) = (A_1 + zA_2)\mathbf{X}(z) + z\mathbf{b}U(z) \] (6)

\[ \mathbf{Y}(z) = \mathbf{c}^T \mathbf{X}(z) + dU(z) \] (7)

where \( A_1 \) and \( A_2 \) are constant matrices, i.e. they do not contain the variable \( z \). The matrix \( A_1 \) includes the delayed terms while the \( A_2 \) matrix contains the delay-free terms, which in terms of a SC implementation corresponds to delay and delay-free integrators, respectively. For a third order example the final set of matrices take the form,

\[ A_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0 & -a_{12} & 0 \\ a_{21} & 0 & a_{23} \\ 0 & a_{31} & -a_{33} \end{bmatrix} \]

\[ \mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ -b_2 \end{bmatrix}, \quad \mathbf{c}^T = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}, \quad d = d \]

A PCA is an array of capacitors which allows a user to select a capacitor value using digital control.
where \( |a_{11}| = |a_{22}| = |a_0| \), or, in general, \( |a_{(i+1)i}| = |a_{(i+1)i}| = |a_0| \), and \( |a_N| = |a_N| \).

In Figure 1, the third order LDI-based quasi-orthonormal SC filter is shown. Six non-overlapping clock phases are used for multiplexing, described in the next section. The coefficients \( c \) and \( d \) can have either a positive or a negative value, implemented by using either the parenthesis or non-parenthesis clocking scheme respectively.

To obtain the filter coefficients for a given transfer-function, we rearrange (6) and (7) as,

\[
\begin{bmatrix}
  zI - I - (A_1 + zA_2) & 0 \\
  -c^T & 1
\end{bmatrix}
\begin{bmatrix}
  X(z) \\
  Y(z)
\end{bmatrix}
= \begin{bmatrix}
  zb \\
d
\end{bmatrix} U(z)
\]

and using Cramer's rule, the transfer-function, \( T(z) \), of the state-space system is given by,

\[
T(z) = \frac{Y(z)}{U(z)} = \frac{p(z)}{c(z)} = \frac{\det \left( (zI - (A_1 + zA_2)) zb \right)}{\det \left( (zI - (A_1 + zA_2)) d \right)}
\]

To solve Equation (8) for the filter coefficients, a symbolic mathematical manipulator, MAPLE [9] is used.

![Diagram of a third-order LDI-based quasi-orthonormal switched-capacitor filter.](image)

**Figure 1:** A third-order LDI-based quasi-orthonormal switched-capacitor filter.

### 2.2 Multiplexing

The time-domain form of Equations (6) and (7) best illustrates the concept of multiplexing this structure. Taking the inverse z-transform of (6) and (7) results in the following time-domain expressions,

\[
x_i(k) = x_i(k-1) + \sum_{j=1}^{N} a_{ij} x_j(k-1) + \sum_{j=1}^{N} b_{ij} x_j(k) + d_i u(k),
\]

\( i = 1, \cdots, N \)  \hspace{1cm} (9)

In Figure 2 (a) a signal-flow graph which realizes these equations for the LDI-based quasi-orthonormal state-space system is shown. In Equations (9) and (10) each intermediate state \( x_i(k) \) is the sum of it's past state \( x_i(k-1) \), and a weighted sum of some of the other states. The past states are obtained by a delay \( z^{-1} \), while the present states are fed in without a delay. For the third order filter, the even ordered state, \( x_2(k) \) uses a weighted sum of it's past state, \( x_2(k-1) \), and the past odd ordered states. The odd ordered states \( x_1(k) \) and \( x_3(k) \), use a weighted sum of the present even ordered state and it's own past state. The three intermediate state equations used in a third-order example.

\[
x_1(k) = x_1(k-1) + a_{11} x_1(k) + a_{12} x_2(k)
\]

\[
x_2(k) = x_2(k-1) + a_{21} x_1(k-1) + a_{22} x_2(k-1) + a_{23} x_3(k-1)
\]

\[
x_3(k) = x_3(k-1) + a_{31} x_1(k) + a_{32} x_2(k) + a_{33} x_3(k) + b u(k)
\]

Note that the first state, \( x_1(k) \) has only one even ordered present state and that the last state is the sum of only one even present state, \( x_3(k) \), a feedback of its own present state and the input. In general, for an odd order filter the even ordered states will depend on \( A_1 \) only, while the odd ordered states will depend on \( A_2 \) only. The reverse is true for an even order filter. Equation (10) for the 3rd order example is,

\[
y(k) = c_1 x_1(k) + c_2 x_2(k) + c_3 x_3(k) + d u(k)
\]

From Equations (11) to (14) we see that there are 10 feed-in capacitors (coefficients) in the whole circuit. In order for the circuit to be fully programmable, arbitrary transfer functions can be realized without changing \( b_3 \). Therefore, we will exclude \( b_3 \) from our list of coefficients to be multiplexed.

![Diagram of a LDI-based state-space structure.](image)

**Figure 2:** Diagram of a LDI-based state-space structure.

\(^2\)However, in some filter transfer functions, a low-precision PCA might be used for \( b_3 \) to improve the dynamic range performance.
Let us see why we choose 3 PCAs to multiplex the coefficients. To calculate a state (see Equations (11) - (13)), the maximum number of coefficients to be multiplexed are 2 (excluding \( h_0 \)). Let us look at state \( x_2(k) \). It is the only state which is a weighted sum of its own present state, i.e. \( a_{33} \) is in the feed-back loop of the op-amp or, is the terminating capacitor (see Figure (1)). This implies that when state \( x_2(k) \) is calculated, \( a_{33} \) should be present. In general, the state \( x_N(k) \) is the only state which has a weighted feed-in of its own present state. Since \( x_2(k) \) depends on \( a_{32} \) and \( a_{33} \) we need at least 2 PCAs, that is, if \( h_0 \) is kept constant.

Therefore, to realize an \( N \)th order fully-programmable SC filter, the minimum number of PCAs is 2. However, there is a trade off between the number of PCAs used and the number of clock-phases required. The relationship between the number of clock-phases, coefficients to be multiplexed and the PCAs is given by:

\[
\text{no. of clock phases} = 2 \times \left( \frac{\text{no. of multiplexed coefficients}}{\text{no. of PCAs}} \right)
\]  

(15)

Using the expression in Equation (15), if we choose 2 PCAs to multiplex an odd, \( N \)th order filter we need \( 3N+1 \) clock phases (10 for \( N=3 \)). This means that, to realize higher order filters a large number of clock-phases are needed. This in turn means shorter charging times for the feed-in capacitors and the requirement for fast op-amps. Therefore, to realize the fully-programmable SC filter for \( N \leq 5 \), 3 PCAs and 2N clock phases will be used. For \( N > 5 \), more than 3 PCAs may be used.

In this section, a step-by-step functional description of the multiplexing scheme is presented. To understand the multiplexing scheme, the six clock-phases are paired into 3 intervals by \( \phi_{iA} - \phi_{iB} \), where \( i = 1, 2, 3 \). This is shown in Figure (3). During each interval, three coefficients are serviced by the PCAs.

\[
x_1(k) = x_1(k-1) + a_{21}x_2(k)
\]

\[
x_2(k) = x_2(k-1) + a_{32}x_3(k) + a_{33}y(k)
\]

During the first interval, the state \( x_2(k) \) is calculated using the odd-ordered past states \( x_1(k) \) and \( x_3(k) \). Since there are 3 PCAs and only 2 of them are used for \( x_2(k) \), we use the third PCA to get the weighted sum of the input. Note that during the first clock phase \( \phi_{1A} \) the output summer is cleared, thus a new sum is calculated during each clock period.

\[
x_3(k) = x_3(k-1) + a_{22}x_2(k) + a_{32}x_3(k) + h_0U(k)
\]

(16)

The odd ordered states are calculated during the second interval. The capacitor related to the coefficient value \( h_0 \) is a fixed capacitor, and therefore there are four weighted sums being calculated during this interval.

\[
y_1(k) = y_1(k) + c_1x_1(k) + c_2x_2(k) + c_3x_3(k) + y, \quad \text{ during } \phi_{2A} \]

(17)

\[
y_2(k) = y_2(k) \quad \text{ during } \phi_{2B} \]

(18)

In the third interval the weighted sum of all the states are calculated. Note that the weighted sum of the input was already calculated during \( \phi_{1A} \). The output is sampled-and-held during \( \phi_{3B} \).

3 Prototype Results

A discrete prototype for \( N \leq 5 \) was built. Memory locations in EPROM were assigned for each PCA. Coefficients for five transfer functions were programmed into these memories and the results are shown in Figure (3a) to (3e). Note in Figure (3a) a transmission zero is shown at half the sampling rate since an exact design technique was used. In the 5th order allpass filter, (Figure (3e)) the small glitch at 1 kHz is due to the inexact matching of the poles and zeros at the passband edge. Also note that all filters have a droop due to the sample-and-hold inherent in the filter.

4 Summary and Conclusion

The goal of this paper was to develop an approach to realize fully-programmable SC filters of higher (> 2) order. The proposed approach implements the filter coefficients using time-shared PCAs. The LDI-based state-space structure used for multiplexing have no delay-free (or continuous) loops and thus do not require unswitched capacitors. Also, this structure is sparse, and therefore very few capacitor values determine the filter's transfer function. The LDI-based quasi-orthonormal structure also supports arbitrary transfer functions such as bandpass, highpass, allpass etc, making it attractive for the programmable filter.

\( \text{---} \)

3 If \( b_0 \) is not kept constant, we need at least 3 PCAs.

\( ^bC_0 \) is shorted
Figure 4: Output of the prototype filter.

References


