CANONIC LOOK AHEAD: CRITICAL CYCLE RELAXED IIR FILTERING WITH MINIMUM MULTIPLICATIVE COMPLEXITY.

J.Rubio,J.Sala,F.Núnez

Department of Signal Theory and Communications. Universitat Politècnica de Catalunya.
Campus Nord, Ed. D5, Jordi Girona 1 i 3, 08034, Barcelona (Spain).
Email: {alvarez,jrubio}@gps.tsc.upc.es

ABSTRACT

In this paper we present an architecture to relax the critical cycle associated with the feedback operations in IIR filtering when high sampling frequencies exceed the computation bandwidth of digital arithmetics. Its complexity, evaluated in terms of multiplications per output sample per pole, equals that of the canonic IIR recursion and provides considerable savings in comparison to existing techniques such as Clustered and Scattered Look Ahead. The procedure is shown to yield unconditionally stable implementations.

1. INTRODUCTION

The computation of very high speed digital FIR filters is accomplished via parallelization and pipelining of operations. Nevertheless, the feedback inherent in IIR filters makes its computation not so straightforward a task. The latency associated with arithmetic operations, and in particular, with the feedback path, limits the maximum sampling frequency of the input signal. Architectures other than the conventional tap-delay filtering have been extensively sought, of which Clustered (CLA) [1], Scattered Look Ahead (SLA) [2],[4] and Minimum Denominator Multiplying (MDM) [3] constitute the best known. [5] and [6] are also references of interest. The philosophy is to modify the numerator and denominator of the filter $H_{z^{-1}} = N_{z^{-1}}/D_{z^{-1}}$ so that the same response $H_{z^{-1}} = N'_{z^{-1}}/D'_{z^{-1}}$ is preserved but the computation of $1/D'_{z^{-1}}$ can be tackled at the operation rate. By way of an example, SLA would perform the following transformation on a single real pole filter,

$$H_{z^{-1}} = \frac{N_{z^{-1}}}{1-\rho z^{-1}} \overset{(\mathrm{SLA})}{\rightarrow} H_{z^{-1}} = \frac{N_{z^{-1}} M_{z^{-1}}}{1-\rho^m z^{-m}} \qquad (1)$$

with $m = 2^r$ and $M_{z^{-1}} = \Pi_{i=0}^{r-1}(1+\rho^{2^i} z^{-2^i})$, so that the zeroes of the polynomial $M_{z^{-1}}$ would cancel out the additional (stable) poles introduced in the denominator. The computation of the filter output at the sampling frequency can now be guaranteed as the computation of the feedback can take place at the operation frequency. The computation of the denominator filter can always be efficiently parallelized as no feedback occurs. The suite of techniques proposed in the literature to include additional poles while maintaining filter stability do always incur in additional complexity. The multiplicative complexity of these architectures, $C_\times$, defined as multiplies per sample per pole, always increases with respect to the complexity of the conventional IIR filter architecture. SLA provides an increase of $r = \log_2 m$ times in multiplicative complexity.

The technique we propose, Canonic Look Ahead (CaLA), provides considerable savings in $C_\times$ with respect to previously known critical cycle relaxing architectures, so that the final complexity in the implementation of any IIR filter equals that of the canonic IIR recursion. $C_\times$ is bounded in all cases by,

$$C^{[\mathrm{CaLA}]}_{\times,\min} = 1 < C^{[\mathrm{CaLA}]}_{\times} \leq C^{[\mathrm{CaLA}]}_{\times,\max} = 2 - m^{-1} \qquad (2)$$

where $C^{[\mathrm{CaLA}]}_{\times,\min}$ would be reached by a IIR filter with an infinite number of poles and $C^{[\mathrm{CaLA}]}_{\times,\max}$ constitutes the multiplicative complexity of a single pole filter. Let P denote the number of poles. Then, $C^{[\mathrm{CaLA}]}_{\times,P+1} < C^{[\mathrm{CaLA}]}_{\times,P}$ monotonically. Thus, CaLA approaches the complexity of the conventional IIR architecture asymptotically. This is done at the expense of the additive complexity C_+, which increases by a factor of $2 - m^{-1}$ with respect to the conventional architecture, irrespectively of the number of poles P.

We will describe CaLA in terms of first and second order stages to implement real pole factors and conjugate pair pole factors in the denominator $D_{z^{-1}}$. The minimum multiplicative complexity is achieved by CaLA through exploitation of the common operations in computing the filter $M_{z^{-1}}$. Therefore, CaLA is essentially a block processing or polyphase scheme. The parallelization of the direct numerator filter $N_{z^{-1}}$ constitutes a different problem and is not considered in the scope of this paper. Also, complexity is only analyzed in terms of number of operations (multiplications and additions). More detailed analyses should consider bit-level implementation for a given application: the dynamic range and bit width of signals.

2. CANONIC LOOK AHEAD

We will consider the critical cycle relaxed implementation of first and second order stages of the objective IIR fil-

0
This work was supported by TIC98-0412, TIC98-0703, TIC99-0849 (CICYT) and CIRIT/Generalitat de Catalunya 1998SGR-00081.

ter separately in the following sub-sections. Multiplicative complexity of each derived architecture is analyzed later and shown to equal that of the canonic update recursion.

2.1. Real Poles

Let us consider the implementation of the single real pole filter $H_{i,z^{-1}} = \left(1 - \rho_i z^{-1}\right)^{-1}$. The corresponding time update equation is given by,

$$y_k = x_k + \rho_i y_{k-1} \tag{3}$$

with x_k and y_k the input and output of the stage $H_{i,z^{-1}}$, respectively. We will refer to (3) as the conventional or canonic time update equation. We define the following vectors,

$$\begin{aligned} \mathbf{y}_k &= \left[y_k, y_{k-1}, \cdots, y_{k-m+1}\right]^{\mathrm{T}} \qquad (4) \\ \mathbf{x}_k &= \left[x_k, x_{k-1}, \cdots, x_{k-m+1}\right]^{\mathrm{T}} \end{aligned}$$

Then, the recursion in (3) can be expressed as,

$$\begin{aligned} \mathbf{y}_k &= \mathbf{C}_i \mathbf{x}_k + \mathbf{D}_i \mathbf{1} y_{k-m} \qquad (5) \\ \mathbf{C}_i &= \begin{bmatrix} 1 & \rho_i & & \rho_i^{m-1} \\ 0 & 1 & & \\ & & \ddots & \rho_i \\ 0 & & 0 & 1 \end{bmatrix}, \\ \left[\mathbf{C}_i\right]_{p,q} &= \begin{array}{ll} \rho_i^{q-p} & , \; p \le q \\ 0 & , \; p > q \end{array} \\ \left[\mathbf{D}_i\right]_{p,q} &= \begin{array}{ll} \rho_i^{m+1-p} & , \; p = q \\ 0 & , \; p \ne q \end{array} \\ \mathbf{1}^{\mathrm{T}} y_{k-m} &= y_{k-m}\left[1, 1, \cdots, 1\right]^{\mathrm{T}} \end{aligned}$$

with the indices fulfilling $1 \le p, q \le m$, $\mathbf{C}_i$ upper triangular and $\mathbf{D}_i$ diagonal. We re-express (5) in terms of the new diagonal matrix $\mathbf{\Delta}_i$ defined by $\left[\mathbf{\Delta}_i\right]_{p,p} = \rho_i^{p-1}$. Hence,

$$\mathbf{D}_i \mathbf{\Delta}_i = \rho_i^m \mathbf{I} \tag{6}$$

with $\mathbf{I}$ the $m \times m$ identity matrix. Then,

$$\mathbf{y}_k = \mathbf{C}_i \mathbf{x}_k + \mathbf{\Delta}_i^{-1} \mathbf{1} \rho_i^m y_{k-m} \tag{7}$$

Also,

$$\begin{aligned} \mathbf{C}_i &= \mathbf{\Delta}_i^{-1} \mathbf{\Sigma} \mathbf{\Delta}_i \qquad (8) \\ \left[\mathbf{\Sigma}\right]_{p,q} &= \begin{array}{ll} 1 & , \; p \le q \\ 0 & , \; p > q \end{array} \end{aligned}$$

with $\mathbf{\Sigma}$ the upper triangular addition matrix. Then, substitution of (8) into (7) yields,

$$\begin{aligned} \mathbf{y}_k &= \mathbf{\Delta}_i^{-1} \mathbf{\Sigma} \mathbf{\Delta}_i \mathbf{x}_k + \mathbf{\Delta}_i^{-1} \mathbf{1} \rho_i^m y_{k-m} \qquad (9) \\ &= \mathbf{\Delta}_i^{-1} \left(\mathbf{\Sigma} \mathbf{\Delta}_i \mathbf{x}_k + \mathbf{1} \rho_i^m y_{k-m}\right) \end{aligned}$$

Defining matrix $\mathbf{U}_1$ as that having its first column equal to all ones and the rest to zero, we can formulate (9) as,

$$\begin{aligned} \mathbf{y}_k &= \mathbf{\Delta}_i^{-1} \left(\mathbf{\Sigma} \mathbf{\Delta}_i \mathbf{x}_k + \rho_i^m \mathbf{U}_1 \mathbf{y}_{k-m}\right) \qquad (10) \\ \mathbf{y}_k^{\Delta} &= \mathbf{\Delta}_i \mathbf{y}_k \\ \mathbf{y}_k^{\Delta} &= \mathbf{\Sigma} \mathbf{\Delta}_i \mathbf{x}_k + \rho_i^m \mathbf{U}_1 \mathbf{\Delta}_i^{-1} \mathbf{y}_{k-m}^{\Delta} \\ &= \mathbf{\Sigma} \mathbf{\Delta}_i \mathbf{x}_k + \rho_i^m \mathbf{U}_1 \mathbf{y}_{k-m}^{\Delta} \Leftarrow \qquad (11) \\ & \boxed{\mathbf{y}_k = \mathbf{\Delta}_i^{-1} \mathbf{y}_k^{\Delta}} \end{aligned}$$

where $\mathbf{U}_1 \mathbf{\Delta}_i^{-1} = \mathbf{U}_1$. The arrow in (11) indicates the definitive update procedure to compute the i-th filtering stage with minimum multiplicative complexity. Note that the operation count is low as $\mathbf{\Delta}_i$ is diagonal and the product with the upper triangular summation matrix $\mathbf{\Sigma}$ can be efficiently performed with additive complexity equal to m. The corresponding z-transform equivalence can be derived as follows,

$$\begin{aligned} \mathbf{Y}_{z^{-1}}^{\Delta} &= \mathbf{\Sigma} \mathbf{\Delta}_i \mathbf{X}_{z^{-1}} + \rho_i^m z^{-m} \mathbf{U}_1 \mathbf{Y}_{z^{-1}}^{\Delta} \qquad (12) \\ \mathbf{Y}_{z^{-1}}^{\Delta} &= \left(\mathbf{I} - \rho_i^m z^{-m} \mathbf{U}_1\right)^{-1} \mathbf{\Sigma} \mathbf{\Delta}_i \mathbf{X}_{z^{-1}} \\ \mathbf{Y}_{z^{-1}} &= \mathbf{\Delta}_i^{-1} \mathbf{Y}_{z^{-1}}^{\Delta} = \mathbf{H}_{i,z^{-1}} \mathbf{X}_{z^{-1}} \end{aligned}$$

With ,

$$\mathbf{H}_{i,z^{-1}} = \mathbf{\Delta}_i^{-1} \left(\mathbf{I} - \rho_i^m z^{-m} \mathbf{U}_1\right)^{-1} \mathbf{\Sigma} \mathbf{\Delta}_i \tag{13}$$

we obtain the matrix transfer function $\mathbf{H}_{i,z^{-1}}$describing the architecture. It is easy to show that this architecture is unconditionally stable by substituting the inverse of $\mathbf{I} - \rho_i^m z^{-m} \mathbf{U}_1$ in (13),

$$\boxed{\mathbf{H}_{i,z^{-1}} = \mathbf{\Delta}_i^{-1} \left(\mathbf{I} + \frac{\rho_i^m z^{-m}}{1 - \rho_i^m z^{-m}} \mathbf{U}_1\right) \mathbf{\Sigma} \mathbf{\Delta}_i} \tag{14}$$

Hence, the multiplication latency $T_\times$ can be relaxed from $T_\times < T_s$ to $T_\times < mT_s$ as decimation has been effected and the slow feedback loop can tackle the high data sampling frequency $f_s = 1/T_s$. Henceforth, m will be referred to as the latency expansion factor. The remaining matrix operations will introduce additional latency in the final computation of the filter output but are critical cycle free due to their non-recursive nature. The filter will be implemented with the concatenation of the matrix transfer functions of each stage as,

$$\mathbf{Y}_{P,z^{-1}} = \mathbf{H}_{P,z^{-1}} \mathbf{Y}_{P-1,z^{-1}} = \left(\Pi_{i=P}^{1} \mathbf{H}_{i,z^{-1}}\right) \mathbf{X}_{z^{-1}} \tag{15}$$

with $\mathbf{Y}_{P,z^{-1}}$ denoting the vector z-transform at the output of the P-th stage. Additional savings in multiplicative complexity occur when the $\mathbf{\Delta}_i$ of one stage is merged with the $\mathbf{\Delta}_{i-1}^{-1}$ of the previous stage yielding $\mathbf{\Delta}_{i,i-1} = \mathbf{\Delta}_i \mathbf{\Delta}_{i-1}^{-1}$ Let us consider the case $P = 2$ by way of an example,

$$\begin{aligned} \Pi_{i=2}^{1} \mathbf{H}_{i,z^{-1}} &= \mathbf{\Delta}_2^{-1} \left(\mathbf{I} - \rho_2^m z^{-m} \mathbf{U}_1\right)^{-1} \mathbf{\Sigma} \cdot \qquad (16) \\ & \quad \mathbf{\Delta}_{2,1} \left(\mathbf{I} - \rho_1^m z^{-m} \mathbf{U}_1\right)^{-1} \mathbf{\Sigma} \mathbf{\Delta}_1 \\ \mathbf{\Delta}_{2,1} &= \mathbf{\Delta}_2 \mathbf{\Delta}_1^{-1} \end{aligned}$$

with $\mathbf{\Delta}_{2,1}$ also diagonal. The number of multiplies per sample per pole after the $\mathbf{\Delta}_i \mathbf{\Delta}_{i-1}^{-1}$ merging operation are (assuming different real poles),

$$\begin{aligned} C_\times^{[\mathrm{CaLA}]} &= \frac{1}{mP} \left((m-1) + \sum_{i=1}^{P} \left[\overset{\mathbf{\Delta}_i}{(m-1)} + \overset{\rho_i^m z^{-m}}{1} \right] \right) \\ &= 1 + \frac{1}{P} \left(1 - \frac{1}{m}\right) \qquad (17) \end{aligned}$$

where the superscripts $\mathbf{\Delta}_i$ and $\rho_i^m z^{-m}$ indicate what matrix operation in (11) is being considered (trivial multiplications by zero and one are not computed in the evaluation of the multiplicative cost). In comparison, $C_{\times}^{[\mathrm{Ca}]} = 1$ is the multiplicative complexity of the canonic time update recursion. The number of additions per sample in (11) per pole is given by,

$$\begin{aligned} C_{+}^{[\mathrm{CaLA}]} &= \frac{1}{mP}\sum_{i=1}^{P}\left[\overset{\boldsymbol{\Sigma}}{(m-1)} + \overset{\rho_i^m z^{-m}}{m}\right] \qquad (18) \\ &= 2 - m^{-1} \end{aligned}$$

in comparison to $C_{+}^{[\mathrm{Ca}]} = 1$, the additive complexity of the canonic time update recursion. We have assumed that all the addition operations involved by the multiplication by $\boldsymbol{\Sigma}$ are carried out recursively in only $m-1$ additions. When the more general recursion $y_k = \alpha x_k + \rho_i y_{k-1}$ with input amplification α is considered, the input factor α can be merged into $\boldsymbol{\Delta}_i$ in (14) so that complexities become,

$$C_{\times}^{[\mathrm{CaLA}]} = 1 + P^{-1}\ ,\ C_{\times}^{[\mathrm{Ca}]} = 1 + P^{-1}$$

that is, Canonic Look Ahead requires the same multiplicative complexity of the canonic time update recursion, irrespectively of the expansion factor m.

2.2. Conjugate poles

The structure derived for real poles can also be applied to filters possessing conjugate poles. Nonetheless, the single pole sections would require complex arithmetic to the detriment of multiplicative complexity. It is more advantageous to implement conjugate poles with an alternative architecture on real arithmetic. Let us now consider the implementation of the single conjugate pair pole filtering stage $H_{i,z^{-1}} = \left(1-\rho_i z^{-1}\right)^{-1}\left(1-\rho_i^* z^{-1}\right)^{-1}$. The denominator can be expressed as,

$$\begin{aligned} H_{i,z^{-1}}^{-1} &= 1 - 2\,\mathrm{Re}\left[\rho_i\right] z^{-1} + |\rho_i|^2 z^{-2} \\ &= 1 - 2 r_i \zeta_i z^{-1} + r_i^2 z^{-2} \end{aligned}$$

with $r_i = |\rho_i| < 1$ for stability and $0 \le |\zeta_i| \le 1$. The associated time recursion is given by,

$$y_k = x_k + 2 r_i \zeta_i y_{k-1} - r_i^2 y_{k-2} \qquad (19)$$

To arrive at equations similar to (10), we previously pack the y_k's and x_k's into two-component subvectors as,

$$\mathbf{y}'_k = \left[y_k, y_{k-1}\right]^{\mathrm{T}}\ ,\ \mathbf{x}'_k = \left[x_k, x_{k-1}\right]^{\mathrm{T}}$$

The recursion in (19) can now be cast into matrix form as,

$$\begin{aligned} \mathbf{y}'_k &= \mathbf{A}'_i \mathbf{x}'_k + \mathbf{B}'_i \mathbf{y}'_{k-2} \\ \mathbf{A}'_i &= \begin{bmatrix} 1 & 2 r_i \zeta_i \\ 0 & 1 \end{bmatrix}\ ,\ \mathbf{B}'_i = \mathbf{A}'_i \begin{bmatrix} -r_i^2 & 0 \\ 2 r_i \zeta_i & -r_i^2 \end{bmatrix} \end{aligned}$$

This is also an unconditionally stable recursion as the eigenvalues of $\mathbf{B}'_i$ have modulus $r_i < 1$,

$$\lambda(\mathbf{B}'_i) = r_i^2\left(2\zeta_i^2 - 1 \pm j\sqrt{1-\left(2\zeta_i^2-1\right)^2}\right)$$

It only remains now to re-vectorize data as,

$$\begin{aligned} \mathbf{y}_k^{\mathrm{T}} &= \left[\mathbf{y}_k'^{\mathrm{T}}, \mathbf{y}_{k-2}'^{\mathrm{T}}, \cdots, \mathbf{y}_{k-2m'}'^{\mathrm{T}}\right] \\ \mathbf{x}_k^{\mathrm{T}} &= \left[\mathbf{x}_k'^{\mathrm{T}}, \mathbf{x}_{k-2}'^{\mathrm{T}}, \cdots, \mathbf{x}_{k-2m'}'^{\mathrm{T}}\right] \end{aligned}$$

with $m = 2m'$. Then, we are able to write,

$$\begin{aligned} \mathbf{y}_k &= \mathbf{C}_i \mathbf{x}_k + \mathbf{D}_i \mathbf{1}' \mathbf{y}'_{k-m} \qquad (20) \\ \mathbf{C}_i &= \begin{bmatrix} \mathbf{A}'_i & \mathbf{B}'_i\mathbf{A}'_i & & \mathbf{B}_i'^{m'-1}\mathbf{A}'_i \\ \mathbf{0} & \mathbf{A}'_i & & \\ & & \ddots & \mathbf{B}'_i\mathbf{A}'_i \\ \mathbf{0} & & \mathbf{0} & \mathbf{A}'_i \end{bmatrix}, \\ \mathbf{D}_i &= \begin{bmatrix} \mathbf{B}_i'^{m'} & \mathbf{0} & & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_i'^{m'-1} & & \\ & & \ddots & \mathbf{0} \\ \mathbf{0} & & \mathbf{0} & \mathbf{B}'_i \end{bmatrix} \\ \mathbf{1}'^{\mathrm{T}} &= \left[\mathbf{I}_2, \mathbf{I}_2, \cdots, \mathbf{I}_2\right]\ ,\ \mathbf{I}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \end{aligned}$$

Now, Defining matrix $\mathbf{U}'_1$ as that having its first two columns equal to matrix $\mathbf{1}'$ and the rest to zero, we obtain the vectorized form,

$$\mathbf{y}_k = \mathbf{C}_i \mathbf{x}_k + \mathbf{D}_i \mathbf{U}'_1 \mathbf{y}_{k-m}$$

where now, as in (6), $\mathbf{D}_i$ and $\boldsymbol{\Delta}_i$ are both block-diagonal and,

$$\mathbf{D}_i \boldsymbol{\Delta}_i = \begin{bmatrix} \mathbf{B}_i'^{m'} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{B}_i'^{m'} \end{bmatrix} = \mathbf{B}_i^{m'}$$

In its turn, $\mathbf{C}_i$ is now expressed as $\mathbf{C}_i = \boldsymbol{\Delta}_i^{-1}\boldsymbol{\Sigma}\boldsymbol{\Delta}_i\mathbf{A}_i$, with,

$$\mathbf{A}_i = \begin{bmatrix} \mathbf{A}'_i & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}'_i \end{bmatrix}\ ,\ \boldsymbol{\Sigma} = \begin{bmatrix} \mathbf{I}_2 & \mathbf{I}_2 & \mathbf{I}_2 \\ \mathbf{0} & \ddots & \mathbf{I}_2 \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_2 \end{bmatrix}$$

Reproducing the same steps, we arrive at the equivalent expression to that in (10),

$$\begin{aligned} \mathbf{y}_k &= \boldsymbol{\Delta}_i^{-1}\left(\boldsymbol{\Sigma}\boldsymbol{\Delta}_i\mathbf{A}_i\mathbf{x}_k + \boldsymbol{\Delta}_i\mathbf{D}_i\mathbf{U}'_1\mathbf{y}_{k-m}\right) \qquad (21) \\ \mathbf{y}_k^{\Delta} &= \boldsymbol{\Delta}_i \mathbf{y}_k \qquad (22) \\ \mathbf{y}_k^{\Delta} &= \boldsymbol{\Sigma}\boldsymbol{\Delta}_i\mathbf{A}_i\mathbf{x}_k + \mathbf{B}_i^{m'}\mathbf{U}'_1\boldsymbol{\Delta}_i^{-1}\mathbf{y}_{k-m}^{\Delta} \qquad (23) \\ &= \boldsymbol{\Sigma}\boldsymbol{\Delta}_i\mathbf{A}_i\mathbf{x}_k + \mathbf{B}_i^{m'}\mathbf{U}'_1\mathbf{y}_{k-m}^{\Delta} \Leftarrow \qquad (24) \\ & \boxed{\mathbf{y}_k = \boldsymbol{\Delta}_i^{-1}\mathbf{y}_k^{\Delta}} \end{aligned}$$

for one single filtering stage, where $\mathbf{U}'_1\boldsymbol{\Delta}_i^{-1} = \mathbf{U}'_1$. So that with $\boldsymbol{\Delta}_i$ and $\mathbf{A}_i$ block-diagonal themselves, we can merge them into a single block-diagonal matrix. The corresponding z-transform equation is provided by $\mathbf{Y}_{z^{-1}} = \mathbf{H}_{i,z^{-1}}\mathbf{X}_{z^{-1}}$, with,

$$\boxed{\mathbf{H}_{i,z^{-1}} = \boldsymbol{\Delta}_i^{-1}\left(\mathbf{I} - z^{-m}\mathbf{B}_i^{m'}\mathbf{U}'_1\right)^{-1}\boldsymbol{\Sigma}\boldsymbol{\Delta}_i\mathbf{A}_i} \qquad (25)$$

The product $\mathbf{H}_{z^{-1}} = \Pi_{i=P_c}^{1}\mathbf{H}_{i,z^{-1}}$ with $P_c = P/2$ the number of conjugate pole pairs does already produce an efficient matrix transfer function. Nevertheless, slight savings in multiplicative complexity can be achieved after some manipulations: additional algebra shows that,

$$\begin{aligned} &\mathbf{\Delta}_i^{-1}\left(\mathbf{I}-z^{-m}\mathbf{B}_i^{m'}\mathbf{U}_1'\right)^{-1}\mathbf{\Sigma}\mathbf{\Delta}_i\mathbf{A}_i \qquad (26)\\ = \quad &\mathbf{A}_i\mathbf{\Lambda}_i^{-1}\left(\mathbf{I}-z_i^{-m}\mathbf{T}_i^{m'}\mathbf{U}_1'\right)^{-1}\mathbf{\Sigma}\mathbf{\Lambda}_i \end{aligned}$$

with $\mathbf{T}_i = \mathbf{A}_i^{-1}\mathbf{B}_i\mathbf{A}_i$ and $\mathbf{\Lambda}_i = \mathbf{A}_i^{-1}\mathbf{\Delta}_i\mathbf{A}_i$ also block diagonal. Hence, cascading all P_c conjugate pole stages with P_c the number of conjugate pole pairs, allows us to bring all the $\mathbf{A}_i$ matrices to the final stage such that the corresponding multiplies can be merged and overall multiplicative complexity reduced. In view of the previous property, we consider the recursive definition of the auxiliary matrices,

$$\mathbf{K}_i = \Pi_{l=i}^{1}\mathbf{A}_l \ , \ \mathbf{G}_i = \mathbf{K}_i^{-1}\mathbf{B}_i\mathbf{K}_i \ , \ \mathbf{\Lambda}_i = \mathbf{K}_i^{-1}\mathbf{\Delta}_i\mathbf{K}_i \qquad (27)$$

so that the final expression for the global matrix transfer function $\mathbf{H}_{z^{-1}}$ is,

$$\boxed{\mathbf{H}_{z^{-1}} = \left(\Pi_{i=P_c}^{1}\mathbf{A}_i\right)\Pi_{i=P_c}^{1}\mathbf{\Lambda}_i^{-1}\left(\mathbf{I}-z^{-m}\mathbf{G}_i^{m'}\mathbf{U}_1'\right)^{-1}\mathbf{\Sigma}\mathbf{\Lambda}_i}$$

This matrix transfer function defines an alternative time recursion to that in (21) and will also allow multiplication merging between consecutive stages. Resorting to the special structure of the matrices $\mathbf{A}_i'$ and $\mathbf{B}_i'$, the evaluation of the multiplicative complexity of this architecture (represented in the figure) yields,

$$C_\times^{[\mathrm{CaLA}]} = 1 + \frac{1}{P}\left(2 - \frac{1}{m}\right) \ , \ C_\times^{[\mathrm{Ca}]} = 1 + P^{-1}$$

So that for conjugate poles, CaLA is slightly above the multiplicative complexity of the canonic recursion, but approximates unity asymptotically in the number of poles.

3. CONCLUSIONS

An algorithm for IIR filtering using slow arithmetic has been proposed that equals the canonic recursion in multiplicative complexity (multiplies per sample per pole approximately equal to unity). The critical cycle can be expanded to handle high input/output data bandwidths with a smaller operation bandwidth. Additive complexity in the AR polynomial is almost doubled with respect to the canonic recursion. Two architectures for the unconditionally stable implementation of single and conjugate pole stages have been presented. In terms of bit-level complexity, a more detailed analysis is necessary for each particular filter, as the pole distribution will determine the dynamic range and bit widths necessary at each point in the proposed architecture, which will ultimately determine area and power consumption.

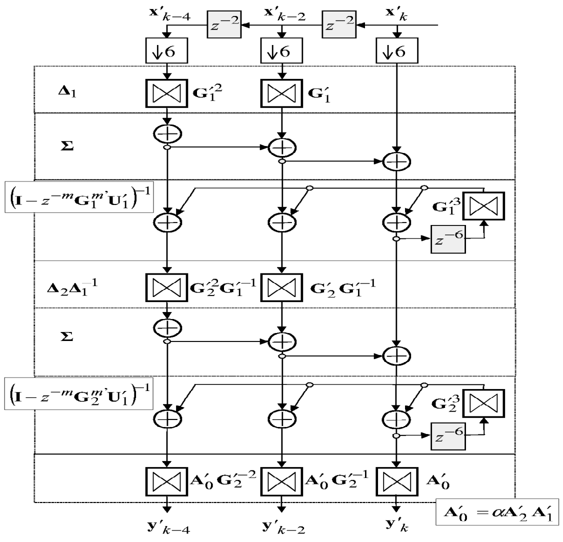
$\mathbf{x}'_{k-4}$
z^{-2}
$\mathbf{x}'_{k-2}$
z^{-2}
$\mathbf{x}'_k$
$\downarrow 6$
$\downarrow 6$
$\downarrow 6$
$\mathbf{\Delta}_1$
$\mathbf{G}_1'^2$
$\mathbf{G}_1'$
$\mathbf{\Sigma}$
$\left(\mathbf{I}-z^{-m}\mathbf{G}_1^{m'}\mathbf{U}_1'\right)^{-1}$
$\mathbf{G}_1'^3$
z^{-6}
$\mathbf{\Delta}_2\mathbf{\Delta}_1^{-1}$
$\mathbf{G}_2'^2\mathbf{G}_1'^{-1}$
$\mathbf{G}_2'\mathbf{G}_1'^{-1}$
$\mathbf{\Sigma}$
$\left(\mathbf{I}-z^{-m}\mathbf{G}_2^{m'}\mathbf{U}_1'\right)^{-1}$
$\mathbf{G}_2'^3$
z^{-6}
$\mathbf{A}_0'\mathbf{G}_2'^{-2}$
$\mathbf{A}_0'\mathbf{G}_2'^{-1}$
$\mathbf{A}_0'$
$\mathbf{A}_0' = \alpha\mathbf{A}_2'\mathbf{A}_1'$
$\mathbf{y}'_{k-4}$
$\mathbf{y}'_{k-2}$
$\mathbf{y}'_k$

Figure 1: $H_{z^{-1}} = \alpha\Pi_{i=1}^{2}\left(1-2r_i\zeta_i z^{-1}+r_i^2z^{-2}\right)^{-1}$with an expansion factor $m = 6$. The scheme is exposed in equation (27). The three vertical data paths are processing two-component vectors and boxes identify the 2×2 matrices derived in the text. The whole data path operates at 1/6 the sampling frequency. The latency of non-recursive operations can be absorbed via pipeline registers which should be placed at the input to each parallel processing block.

4. REFERENCES

[1] Kyung Hi Chang, Improved Clustered Look-Ahead Pipelining Algorithm with Minimum Order Augmentation, IEEE Trans. on Signal Processing, p. 2575-2579, VOL.45, no.10, October 1997.

[2] Keshab K. Pari, David G. Messerschmitt, Pipeline Interleaving and Parallelism in Recursive Digital Filters-Part I: Pipelining Using Scattered Look Ahead and Decomposition, IEEE Trans. on Acoustics, Speech and Signal Processing, p. 1099-1117,VOL. 37, no.7, July 1989.

[3] A.E. de la Serna, M.A. Soderstrand, Canonical MDM Pipelined Digital Filters, Proceedings of the 37th Midwest Symposium on Circuits and Systems (1994), p. 1079-1082, VOL.2.

[4] Gatherer, A., Papamichalis, P., Scattered Lookahead without finding Polynomial Roots or solving Simultaneous Equations, IEEE Transactions on Signal Processing, VOL.47, Issue 9, Sept. 1999, Page(s): 2415-2418.

[5] Miroslav D Lutovac, Dejan V. Tosic, Brian Evans, Filter Design for Signal Processing, Prentice Hall, ISBN 0-201-36130-2, 2001

[6] Keshab Parhi, VLSI Digital Signal Processing Systems, John Wiley and Sons, ISBN 0-471-24186-5, 1999, http://www.ece.umn.edu/users/parhi/