FIXED-POINT IMPLEMENTATION OF LATTICE WAVE DIGITAL FILTER: COMPARISON AND ERROR ANALYSIS

Anastasia Volkova, Thibault Hilaire

Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6, F-75005, Paris, France

ABSTRACT

A consistent analysis of the filter design along with its further implementation in fixed-point arithmetic requires a large amount of work, and this process differs from one filter representation to another. For the unifying purposes of such flow, a Specialized Implicit Form (SIF) had been proposed in [1].Various sensitivity and stability measures have been adapted to it along with an *a priori* error analysis (quantization of the coefficients and output error). In this paper a conversion algorithm for the widely used Lattice Wave Digital Filters (LWDF) to the SIF is presented, along with a finite precision error analysis. It allows to compare fairly LWDF to other structures, like direct forms and state-space. This is illustrated with a numerical example.

***Index Terms*—** Filter implementation, Lattice Wave Digital Filters, error analysis, fixed-point arithmetic.

1. INTRODUCTION

Most of control and signal processing algorithms are implemented for application in embedded systems, which use finite-precision arithmetic. Unfortunately, the quantization of the coefficients and the roundoff errors in the computations lead to degradation of the algorithms. This makes the implementation process tedious and error-prone, since no automatic tool exists for the filter-to-code transformation with a rigorous error analysis. Moreover, the diversity of Infinite Impulse Response (IIR) filter realizations (Direct Forms, state-space, Wave, etc.) must be taken into consideration. These realizations are equivalent mathematically but not anymore in finite-precision arithmetic. Their structures make some of them more *sensitive* to the quantization of the coefficients, whereas some of them are very *robust* to roundoff errors. Furthermore, each representation demands its own, often difficult and time-consuming, analysis procedure.

For the unification of filter analysis, a Specialized Implicit Form (SIF) was introduced in [1]. It allows to represent any filter in a unified form and then to apply a rigorous stability/sensitivity and error analysis, such as transfer function and pole sensitivity measures or output error. Error analysis is based on rigorous algorithms, which compute error intervals for an implemented filter. Moreover, automatic Fixed-Point code generation tool for the SIF was proposed in [2] and is based on particular implementation of Sum-of-Products (SoPs).

The complete flow of filter implementation using SIF is shown in Figure 1. Since this process is unified, other realizations of the same filter may be simultaneously considered and a fair comparison can be provided.

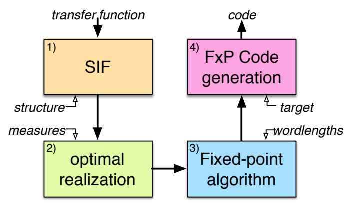


Fig. 1: From transfer function to Fixed-Point code.

In this article, we consider Lattice Wave Digital Filters (LWDF) as a target filter realization, which we convert to SIF. We show some of the filter analysis measures available for SIF and apply them on LWDF for further comparison with the results of Direct Form I (DFI), ρ-Direct Form II transposed (ρDFIIt [3]) and state-space realizations.

For that purpose, LWDF are recalled in Section 2. A fixed-point arithmetic error analysis for the SIF is detailed in Section 3 and a LWDF-to-SIF conversion algorithm is exhibited in Section 4. Finally, a comparative example is given.

Notation: vectors are in lowercase bold, matrices are in uppercase bold. All matrix inequalities are applied element-by-element. Operator $\times$ denotes an entrywise matrix product; $\lfloor x \rceil_2$ is the nearest power of 2 lower than x.

2. LATTICE WAVE DIGITAL FILTERS

LWDF is a class of recursive Wave Digital Filters that inherit several good properties, such as stability for implementation and possibility of suppression of parasitic oscillations. LWDF can be either derived from analog reference filters [4] or using explicit formulas [5].

The LWDF structure is highly modular and has a high degree of parallelism, which makes them suitable for a VLSI implementation. Their good stability qualities [4] make them good candidates for adaptive filtering and Hilbert transformers design [6].

This work has been sponsored by french ANR agency under grant No ANR-13-INSE-0007.

LWDF is represented by two parallel branches, which realize all-pass filters. These all-pass filters are composed of cascaded first- and second-order symmetric two-port adaptors. Its data-flow diagram (DFG) is shown on Figure 2.

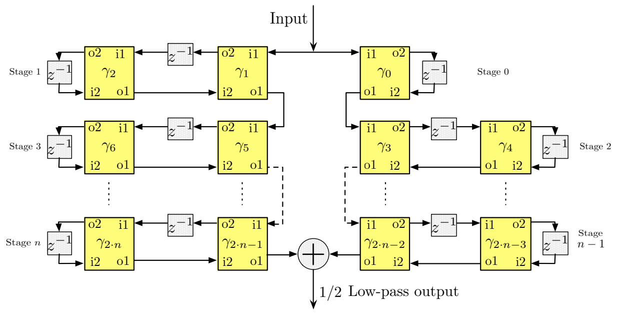


Fig. 2: Data-flow diagram of a low-pass LWDF.

Each adaptor contains three adders and one multiplier. According to [5], the adaptor coefficients γ may be guaranteed to fall into the interval $-1 < \gamma < 1$. In [4] it was proposed to use Richards' structures for adaptors, as on Figure 3.

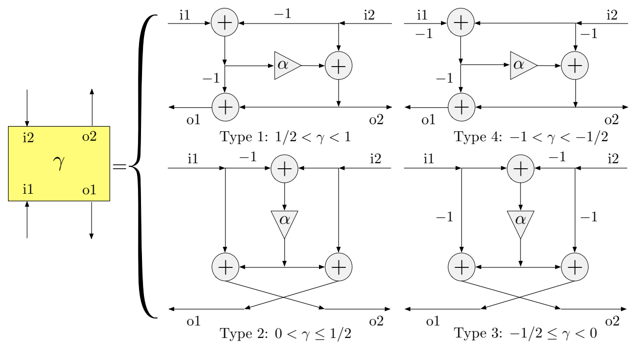

i1
i2
o1
o2
γ
-1
α
Type 1: $1/2 < \gamma < 1$
Type 4: $-1 < \gamma < -1/2$
Type 2: $0 < \gamma \le 1/2$
Type 3: $-1/2 \le \gamma < 0$

Fig. 3: Two-port adaptor structures, for which actual multiplier α is computed out of γ using Table 1.

Moreover, instead of multiplication by γ an easy-to-implement multiplication by $0 < \alpha \leqslant 1/2$ takes place in each type of structure, as summarized in the Table 1.

Type	γ range	Value of α
1	$1/2 < \gamma < 1$	$\alpha = 1 - \gamma$
2	$0 < \gamma \leqslant 1/2$	$\alpha = \gamma$
3	$-1/2 \leqslant \gamma < 0$	$\alpha = \lvert\gamma\rvert$
4	$-1 < \gamma < -1/2$	$\alpha = 1 + \gamma$

Table 1: γ to α conversion for different γ ranges.

It was shown in [4] that in order to make sure that only one passband and only one stop-band occur, the orders of the upper and lower branches must differ by one, such that the overall order n of the filter is odd. The high-pass filter may be obtained simultaneously by changing the sign of the all-pass lower branch.

In [5] explicit formulas for LWDF transfer function design for several common filters such as Butterworth, Cauer (elliptic) and Chebyshev were presented. However, LWDF can realize all reference filters. Due to its good qualities, LWDFs are considered in numerous different applications, including studies on linear-phase structures and energy-efficient structures [7]. However, all studies on lattice wave structures implementation in finite word-length arithmetic are performed **a posteriori**, *i.e.* when the implementation parameters are known [8].These are commonly two- or three-step algorithms that propose coefficients quantization scheme based on solving optimization problems for infinite-precision filter models and then adjustment of finite-precision filter.

In this work, however, we apply a more arithmetic approach on LWDF using SIF, which provides **a priori** bounds on the coefficient's quantization errors and on roundoff errors. These error bounds permit different realizations comparison on the filter design stage.

3. FIXED-POINT ARITHMETIC ERROR ANALYSIS

3.1. Specialized Implicit Framework

In order to encompass all the possible realizations for a given transfer function the Specialized Implicit Framework has been proposed in [1]. SIF is an extension of the state-space realization, modified in order to allow chained Sum-of-Products (SoP) operations. All the input-output relationships with delays, computation order, multiplications by constants and additions can be represented with the SIF. This macroscopic description is more suited for the analysis than a graph relationship as it gives direct analytical formula for the finite precision error analysis [1]. We consider here the Single Input Single Output (SISO) filters, but it can be easily extended to Multiple-Inputs Multiple Outputs (MIMO) cases.

Denote $u(k)$ and $y(k)$ the input and output respectively. Variables that are stored from one step to the other are in the state vector $\boldsymbol{x}(k)$, while intermediate results are collected in the vector $\boldsymbol{t}(k)$. Then, the SIF is the following system:

$$\mathcal{H}\begin{cases} \boldsymbol{Jt}(k+1) = & & \boldsymbol{Mx}(k) + \boldsymbol{N}u(k) \\ \boldsymbol{x}(k+1) = \boldsymbol{Kt}(k+1) & + & \boldsymbol{Px}(k) + \boldsymbol{Q}u(k) \\ y(k) = \boldsymbol{Lt}(k+1) & + & \boldsymbol{Rx}(k) + \boldsymbol{S}u(k) \end{cases} \tag{1}$$

Note that $\boldsymbol{J}$ must be lower triangular with 1 on its diagonal, so the first value of $\boldsymbol{t}(k+1)$ is first computed, then the second one is computed using the first and so on. The *implicit* term $\boldsymbol{Jt}(k+1)$ naturally serves for preservation of the computation order specific for each realization. Its transfer function H can be obtained analogously to the state-space realization.

Denote $\boldsymbol{Z}$ as a set of the SIF coefficients:

$$\boldsymbol{Z} \triangleq \begin{pmatrix} -\boldsymbol{J} & \boldsymbol{M} & \boldsymbol{N} \\ \boldsymbol{K} & \boldsymbol{P} & \boldsymbol{Q} \\ \boldsymbol{L} & \boldsymbol{R} & \boldsymbol{S} \end{pmatrix}. \tag{2}$$

Various filter analysis measures [1, 9, 10] have been explicitly introduced for this framework along with rigorous error analysis algorithms, some of which are described further in this section.

3.2. Fixed-Point Arithmetic

In two's complement Fixed-Point (FxP) arithmetic, a FxP number x is represented by

$$x = -2^m x_m + \sum_{i=\ell}^{m-1} 2^i x_i, \tag{3}$$

where x_i is the i^{th} bit of x, and m and ℓ are the *Most Significant Bit* (MSB) and *Least Significant Bit* (LSB) of x.

If a real non null constant c has to be approximated by a w-bit FxP number, its MSB in most of cases is deduced from

$$m = \lfloor \log_2 |c| \rfloor + 1 \tag{4}$$

and then its LSB ℓ is deduced with $\ell = m - w + 1$ (in some special cases eq. (4) may be inaccurate, see [11] for the complete algorithm). If the round-to-nearest mode is chosen for the conversion, then c is approximated with an absolute error Δc such that $|\Delta c| \leqslant 2^{\ell-1}$.

3.3. Coefficient's quantization

Obviously, after implementation the coefficients $\boldsymbol{Z}$ will be modified into $\boldsymbol{Z} + \boldsymbol{\Delta Z}$, where the errors $\boldsymbol{\Delta Z}$ depend on the coefficients values and word-lengths, according to (4). In order to evaluate how the filter characteristics may be modified by the quantization of the coefficients, sensitivity-based measures are usually used [12].

The sensitivity of the transfer function H with respect to the coefficients $\boldsymbol{Z}$ is given by $\frac{\partial H}{\partial \boldsymbol{Z}}$. Analytical form of this measure is usually developed specifically for each realization. However, once obtained for SIF [1], it can be applied to any realization. Unfortunately, this commonly used measure is not fair and does not reflect how the coefficients' quantization changed the transfer function H into $H + \Delta H$ since the absolute error of the coefficients may not all have the same magnitude order.

For that purpose, a *stochastic* sensitivity-based measure has been proposed and developed with respect to FxP considerations in [9]. Here, quantization error $\boldsymbol{\Delta Z}_{ij}$ of the coefficient $\boldsymbol{Z}_{ij}$ is considered as a random variable, uniformly distributed in $[-2^{\ell_{\boldsymbol{Z}_{ij}}-1}; 2^{\ell_{\boldsymbol{Z}_{ij}}-1}]$ where $\ell_{\boldsymbol{Z}_{ij}}$ is the LSB of the $\boldsymbol{Z}_{ij}$. Then, the error transfer function ΔH can be seen as a transfer function with random variables as coefficients. Its second-order moment is defined as

$$\sigma^2_{\Delta H} \triangleq \frac{1}{2\pi} \int_0^{2\pi} E\left\{ \left| \Delta H \left(e^{j\omega} \right) \right|^2 \right\} d\omega, \tag{5}$$

where $E\{.\}$ is the expectation operator. It reflects how much the transfer function H has changed due to the quantization. From [9], it can be evaluated with

$$\sigma^2_{\Delta H} = \left\| \frac{\partial H}{\partial \boldsymbol{Z}} \times \boldsymbol{\Xi} \right\|_2^2 \tag{6}$$

where

$$\boldsymbol{\Xi}_{ij} \triangleq \begin{cases} 0 & \text{if } \boldsymbol{Z}_{ij} \in \{0, \pm 1\} \\ \frac{2^{-w_{\boldsymbol{Z}_{ij}}+1}}{\sqrt{3}} \lfloor \boldsymbol{Z}_{ij} \rfloor_2 & \text{otherwise} \end{cases} \tag{7}$$

and $w_{\boldsymbol{Z}_{ij}}$ is the word-length fixed to represent $\boldsymbol{Z}_{ij}$. Thus, $\sigma^2_{\Delta H}$ captures more information on the transfer function error.

If all the coefficients have the same word-length, then a normalized transfer function error measure [9] is defined by

$$\bar{\sigma}^2_{\Delta H} \triangleq \left\| \frac{\partial H}{\partial \boldsymbol{Z}} \times \lfloor \boldsymbol{Z} \rfloor_2 \right\|_2^2. \tag{8}$$

This normalization is useful for a concrete realization analysis on the design stage, when word-lengths are not known.

The same approach was applied for the poles $\{\boldsymbol{\lambda}_i\}$ of the systems, or more interestingly to their moduli $|\boldsymbol{\lambda}_i|$ in order to ensure filter's stability after quantization. The poles moduli sensitivity (w.r.t. the coefficients) is measured with $\frac{\partial |\boldsymbol{\lambda}_i|}{\partial \boldsymbol{Z}}$ and detailed in [12]. For the same reason as the transfer function sensitivity, it was developed as the second-order moment of the random variables $\Delta |\boldsymbol{\lambda}_i|$ and analogous to (6) the pole error $\sigma^2_{\Delta|\boldsymbol{\lambda}|}$ was introduced.

Likewise to (8) the word-length independent normalized error pole measure $\bar{\sigma}^2_{\Delta|\boldsymbol{\lambda}|}$ may be computed [12].

3.4. Roundoff errors

In addition to the quantization of the coefficients, the *roundoff errors* occur during the computations. The Sum-of-Products operations required to compute each line of (1) cannot be exactly implemented due to the binary-point alignments and the final fixed-point quantizations.

If some extra guard bits are added, then the SoPs can be performed with faithful rounding of the last bit, *i.e.* with an error ε such that $|\varepsilon| \leqslant 2^{\ell}$ where ℓ is the LSB of the result [10].

Considering these errors leads to the following implemented filter $\mathcal{H}^*$:

$$\begin{aligned} \boldsymbol{Jt}^*(k+1) &= \boldsymbol{Mx}^*(k) + \boldsymbol{N}u(k) + \boldsymbol{\varepsilon_t}(k) \\ \boldsymbol{x}^*(k+1) &= \boldsymbol{Kt}^*(k+1) + \boldsymbol{Px}^*(k) + \boldsymbol{Q}u(k) + \boldsymbol{\varepsilon_x}(k) \\ y^*(k) &= \boldsymbol{Lt}^*(k+1) + \boldsymbol{Rx}^*(k) + \boldsymbol{S}u(k) + \varepsilon_y(k) \end{aligned} \tag{9}$$

where $\boldsymbol{\varepsilon_t}(k), \boldsymbol{\varepsilon_x}(k)$ and $\varepsilon_y(k)$ are the roundoff errors due to the FxP computations of all the SoPs.

Performing the difference between an exact (1) and implemented filter (9), the output error $\Delta y(k) \triangleq y^*(k) - y(k)$ can be seen as the output of the roundoff error vector $\boldsymbol{\varepsilon}(k) \triangleq \left(\boldsymbol{\varepsilon_t}^\top(k), \boldsymbol{\varepsilon_x}^\top(k), \varepsilon_y^\top(k) \right)^\top$ through a (MIMO) error filter denoted $\boldsymbol{\mathcal{H}_\varepsilon}$, as shown in Figure 4. $\boldsymbol{\mathcal{H}_\varepsilon}$ can be easily described by state-space representation [11].

The worst possible output error $\|\Delta y\|_\infty \triangleq \sup_{k \geqslant 0} |\Delta y(k)|$ can be deduced from a bound $\|\varepsilon\|_\infty$ on the error vector $\boldsymbol{\varepsilon}(k)$ with:

$$\|\Delta y\|_\infty = \langle\!\langle \boldsymbol{H_\varepsilon} \rangle\!\rangle \, \|\varepsilon\|_\infty, \tag{10}$$

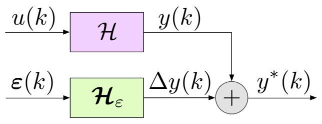


Fig. 4: The implemented filter can be seen as the exact filter perturbed by the roundoff errors.

where $\langle\!\langle \boldsymbol{H}_{\boldsymbol{\varepsilon}} \rangle\!\rangle$ is the Worst Case Peak Gain matrix of the MISO filter $\boldsymbol{\mathcal{H}}_{\boldsymbol{\varepsilon}}$, *i.e.* the L_1-norm of its impulse response. In [13], the authors showed how to evaluate this matrix at any arbitrary precision.

The error vector is bounded by $\|\varepsilon\|_\infty \leqslant 2^{\boldsymbol{\ell_{txy}}}$, where $\boldsymbol{\ell_{txy}}$ is the vector of the LSBs of the variables $\boldsymbol{t}$, $\boldsymbol{x}$ and y. It can be deduced ($\boldsymbol{\ell_{txy}} = \boldsymbol{m_{txy}} - \boldsymbol{w_{txy}} + 1$) from the word-lengths $\boldsymbol{w_{txy}}$ and the MSBs $\boldsymbol{m_{txy}}$ of the variables $\boldsymbol{t}$, $\boldsymbol{x}$ and y.

The word-lengths $\boldsymbol{w_{txy}}$ are an implementation choice (determined by the hardware architecture used for the implementation), whereas $\boldsymbol{m_{txy}}$ may be determined from the bound $\|u\|_\infty$ of the input $u(k)$:

$$\boldsymbol{m_{txy}} = \left\lfloor \log_2 \left(\langle\!\langle \boldsymbol{H}_u \rangle\!\rangle \|u\|_\infty \right) \right\rfloor + 1. \tag{11}$$

where $\boldsymbol{\mathcal{H}}_u$ is the specific filter with takes $u(k)$ as input and returns $\boldsymbol{t}$, $\boldsymbol{x}$ and y, and $\langle\!\langle \boldsymbol{H}_u \rangle\!\rangle$ its Worst-Case Peak-Gain. An explicit state-space form of $\boldsymbol{\mathcal{H}}_u$ can be easily obtained [11].

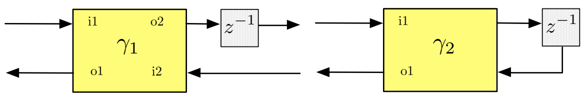
Note that the the Worst-Case Peak-Gain measure used in (11) guaranties that no overflow occurs on $\boldsymbol{t}$, $\boldsymbol{x}$ and y. It is similar to a scaling.

Moreover, if the largest possible value of the input is a power-of-2, then substituting (11) into (10) leads to a normalized output error bound (independent of word-lengths analogously to Section 3.3):

$$\overline{\Delta y} \triangleq \langle\!\langle \boldsymbol{H}_\varepsilon \rangle\!\rangle \left\lfloor \langle\!\langle \boldsymbol{H}_u \rangle\!\rangle \right\rfloor_2 . \tag{12}$$

4. LWDF-TO-SIF CONVERSION ALGORITHM

Due to the high modularity of the LWDF, the conversion between its DFG and SIF is not difficult. As seen on Figure 2, LWDF consists of two branches, and each branch is a cascade of stages. Each stage in its turn may be considered as a cascade of *subsystems* of two types, as shown on Figure 5.



(a) Type A (b) Type B

Fig. 5: A stage is considered as cascade of subsystems of type A (Subfig. a) or type B (Subfig. b).

Therefore, the basic brick for cascade sequence is actually not a simple adaptor but a two-port adaptor with one output delayed (Type A: Figure 5a) and a 1-input/1-output adaptor with delay (Type B: Figure 5b). Then, given filter's coefficients γ, the conversion algorithm can be divided into following steps:

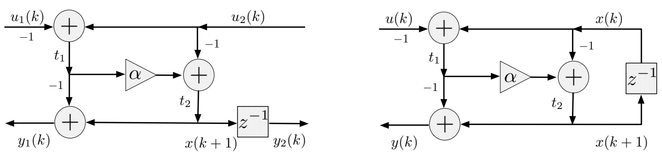
Step 1: Deduce SIF representation (matrix $\boldsymbol{Z}$) for each subsystem according to its γ value;

Step 2: Cascade subsystems into stages;

Step 3: Cascade stages into branches;

Step 4: Regroup two branches SIFs into final filter.

The first step can be easily done by applying SIF notation to the subsystem DFG. Two subsystems per each adaptor structure must be considered, overall 8 basic bricks. For example, for $-1 < \gamma < 1/2$ the structures to be converted into SIF can be described with DFGs on Figure 6.



(a) (b)

Fig. 6: (a) Type A subsystem with adaptor of type 4. (b) Type B subsystem with adaptor of type 4.

Applying SIF definition (1), we can deduce that SIFs $\boldsymbol{Z}_A$ and $\boldsymbol{Z}_B$ for subsystems shown on Figure 6 are:

$$\boldsymbol{Z}_A \triangleq \left(\begin{array}{c:c:c} -\boldsymbol{J}_A & \boldsymbol{M}_A & \boldsymbol{N}_A \\ \hdashline \boldsymbol{K}_A & \boldsymbol{P}_A & \boldsymbol{Q}_A \\ \hdashline \boldsymbol{L}_A & \boldsymbol{R}_A & \boldsymbol{S}_A \end{array}\right) = \left(\begin{array}{cc:c:cc} -1 & 0 & 0 & 1 & 1 \\ \alpha & -1 & 0 & 0 & 1 \\ \hdashline 0 & 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ \hdashline 0 & 0 & 1 & 0 & 0 \end{array}\right) \tag{13}$$

$$\boldsymbol{Z}_B \triangleq \left(\begin{array}{c:c:c} -\boldsymbol{J}_B & \boldsymbol{M}_B & \boldsymbol{N}_B \\ \hdashline \boldsymbol{K}_B & \boldsymbol{P}_B & \boldsymbol{Q}_B \\ \hdashline \boldsymbol{L}_B & \boldsymbol{R}_B & \boldsymbol{S}_B \end{array}\right) = \left(\begin{array}{cc:c:c} -1 & 0 & 1 & -1 \\ \alpha & -1 & 1 & 0 \\ \hdashline 0 & -1 & 0 & 0 \\ \hdashline -1 & 1 & 0 & 0 \end{array}\right) \tag{14}$$

Sequential cascading of two SIFs can also be expressed explicitly. For example, if two SIFs are determined with matrices $\{\boldsymbol{J}_1, \boldsymbol{K}_1, \ldots, \boldsymbol{S}_1\}$ and $\{\boldsymbol{J}_2, \boldsymbol{K}_2, \ldots, \boldsymbol{S}_2\}$ respectively, then the cascaded SIF may be obtained with:

$$\boldsymbol{Z} = \left(\begin{array}{ccc:cc:c} -\boldsymbol{J}_1 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{M}_1 & \boldsymbol{0} & \boldsymbol{N}_1 \\ \boldsymbol{L}_1 & -\boldsymbol{I} & \boldsymbol{0} & \boldsymbol{R}_1 & \boldsymbol{0} & \boldsymbol{S}_1 \\ \boldsymbol{0} & \boldsymbol{N}_2 & -\boldsymbol{J}_2 & \boldsymbol{0} & \boldsymbol{M}_2 & \boldsymbol{0} \\ \hdashline \boldsymbol{K}_1 & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{P}_1 & \boldsymbol{0} & \boldsymbol{Q}_1 \\ \boldsymbol{0} & \boldsymbol{Q}_2 & \boldsymbol{K}_2 & \boldsymbol{0} & \boldsymbol{P}_2 & \boldsymbol{0} \\ \hdashline \boldsymbol{0} & \boldsymbol{S}_2 & \boldsymbol{L}_2 & \boldsymbol{0} & \boldsymbol{R}_2 & \boldsymbol{0} \end{array}\right). \tag{15}$$

Notate, that even on the first step of conversion the matrices $\boldsymbol{Z}_A$ and $\boldsymbol{Z}_B$ are sparse. Each application of cascade formula (15) would produce an even more sparse matrix.

The complete algorithm (for all the subsystems) is not given here dut to lack of space.

5. NUMERICAL EXAMPLES AND COMPARISONS

The following example is based on the LWDF coefficients, which were obtained with Lattice Wave Digital Filters[1] tool-

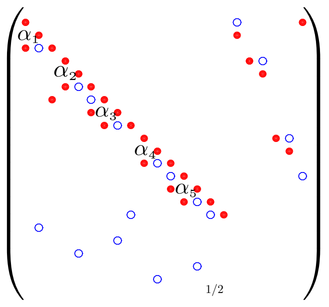
[1]`http://ens.ewi.tudelft.nl/~huib/mtbx/index.php`

box for Matlab. This toolbox is based on explicit formulas introduced in [5]. However, the LWDF-to-SIF conversion algorithm requires only the γs, therefore does not depend on software providing coefficients. The FWR toolbox[2] for Matlab was used for the SIF analysis.

A Matlab-generated low-pass 5^{th} order Butterworth filter with cutoff frequency 0.1 was considered. Four realizations of this transfer function were considered: LWDF, balanced state-space, Direct Form I and ρ Direct Form II transposed (ρDFIIt [3]). Only the normalized (*i.e.* input-width independent) versions of measures introduced in Section 3 were used.

The result SIF for the considered LWDF structure is a sparse 22×22 matrix shown below. It has only three types of non-zero elements: adaptor coefficients α_i and plus/minus ones represented by filled and contour circles respectively:

$\mathbf{Z} =$ (16)



As described in [14, 3], the ρDFIIt is parametrized by n extra parameters. It can be a subject of multi-criteria optimization. For this example a tradeoff function minimizing weighted sum of normalized transfer function and pole errors was used. Excellent results may be observed in Table 2.
However, Direct Form I showed to be very ill-conditioned for the considered filter, and its pole error cannot be computed.

Thanks to the minimal number of coefficients, the LWDF approaches in its normalized transfer function error the optimized ρDFIIt. The specific alternating distribution of LWDF poles [8] leads to good results in pole error measure. Normalized output error is, however, quite large because it was not designed to minimize the propagation of roundoff errors. Therefore, ρDFIIt may be a good alternative. To provide a consistent implementation comparison a code optimization and generation chain FiPoGen [10] + FloPoCo[3] is required.

The state-space realization chosen is a balanced state-space. It is expectantly good at output error, but more sensitive to quantization errors (it has much more coefficients).

6. CONCLUSION

Various studies on LWDF have been introduced over the years. However, existing computational error analysis are dedicated to that particular structure and do not really permit comparisons. The conversion from LWDF to SIF permits to apply numerous classical and novel sensitivity measures upon any LWDF realization and any other.
Further work will consist of using the Fixed-Point Generator [10] capabilities to produce optimal (with respect to either implementation cost and some error criteria) FxP code for various hardware and software architectures.

[2]`https://gforge.inria.fr/projects/fwrtoolbox/`
[3]`http://flopoco.gforge.inria.fr/`

Realization	size(Z)	coeff.	$\bar{\sigma}^2_{\Delta H}$	$\bar{\sigma}^2_{\Delta\lvert\lambda\rvert}$	$\overline{\Delta_y}$
LWDF	22×22	5	0. 3151	0.56	122.9
state-space	6× 6	36	1.15	5.75	23.33
ρDFIIt	11×11	11	0.09	0.45	94.3
DFI	12×12	11	1.42e+6	-	7.961

Table 2: Different realizations comparison.

REFERENCES

[1] T. Hilaire, P. Chevrel, and J.F. Whidborne, "A unifying framework for finite wordlength realizations," *IEEE Trans. on Circuits and Systems*, vol. 8, no. 54, pp. 1765–1774, August 2007.

[2] B. Lopez, *Implémentation optimale de filtres linéaires en arithmétique virgule fixe*, Ph.D. thesis, UPMC, 2015.

[3] Z. Zhao and G. Li, "Roundoff noise analysis of two efficient digital filter structures," *IEEE Transactions on Signal Processing*, vol. 54, no. 2, pp. 790–795, February 2006.

[4] A. Fettweiss, "Wave digital filters: Theory and practice," *Proc. of the IEEE*, vol. 74, no. 2, 1986.

[5] L. Gazsi, "Explicit formulas for lattice wave digital filters," *IEEE Trans. Circuits & Systems*, vol. 32, no. 1, 1985.

[6] H. Johansson and L. Wanharomar, "Digital hilbert transformers composed of identical allpass subfilters," in *ISCAS 1998. Proceedings of*, vol. 5, pp. 437–440 vol.5.

[7] H. Ohlsson O. Gustafsson W. Li L. Wanhammar, "An environment for design and implementation of energy efficient digital filters," in *SSoCC*, apr 2003.

[8] J. Yli-Kaakinen and T. Saramäki, "A systematic algorithm for the design of lattice wave digital filters with short-coefficient wordlength," *IEEE Trans. on Circuits & Systems*, 2007.

[9] T. Hilaire and P. Chevrel, "Sensitivity-based pole and input-output errors of linear filters as indicators of the implementation deterioration in fixed-point context," *EURASIP Journal on Advances in Signal Processing*, January 2011.

[10] B. Lopez, T. Hilaire, and L.-S. Didier, "Formatting bits to better implement signal processing algorithms," in *4th Int. conf. PECCS, proceedings of*, 2014.

[11] T. Hilaire and B. Lopez, "Reliable implementation of linear filters with fixed-point arithmetic," in *IEEE Workshop on Signal Processing Systems, SiPS 2013*, 2013, pp. 401–406.

[12] M. Gevers and G. Li, *Parametrizations in Control, Estimation and Filtering Probems*, Springer-Verlag, 1993.

[13] A. Volkova, T. Hilaire, and C. Lauter, "Reliable evaluation of the Worst-Case Peak Gain matrix in multiple precision," in *IEEE Symposium on Computer Arithmetic*, 2015.

[14] G. Li, C. Wan, and G. Bi, "An improved rho-DFIIt structure for digital filters with minimum roundoff noise," *IEEE Trans. on Circuits & Systems*, vol. 52, no. 4, pp. 199–203, 2005.