IET Circuits, Devices & Systems

Research Article

IET Journals
The Institution of Engineering and Technology

Precision analysis with analytical bit-width optimisation process for linear circuits with feedbacks

ISSN 1751-858X
Received on 4th January 2018
Revised 19th February 2018
Accepted on 5th March 2018
E-First on 15th June 2018
doi: 10.1049/iet-cds.2017.0514
www.ietdl.org

El-Sedik Lamini[1,2] ✉, Samir Tagzout[3], Hacène Belbachir[1], Adel Belouchrani[4]

[1]*USTHB, Faculty of Mathematics, RECITS Laboratory, BP 32, El Alia 16111, Bab Ezzouar, Algiers, Algeria*
[2]*Division de Microélectronique et Nanotechnologie, Centre de Développement des Technologies Avancées, CDTA, Algiers, Algeria*
[3]*Groupe Brandt, Pôle Industrie, CEVITAL, Garidi II, Kouba 16005, Algeirs, Algeria*
[4]*Electrical Engineering Department Ecole Nationale Polytechnique, LDCCP, BP 16028, El Harrach, Algiers, Algeria*
✉ *E-mail: lelsedik@gmail.com*

Abstract: Finding the best possible word length to accuracy trade off seems to be an obvious design task. However, the literature and carful design reviews show that word lengths are often overestimated to put the data accuracy at the safe side. This study proposes a mathematical process to balance that trade off. It describes an analytical optimisation technique that considers every interconnection and it shows clear improvement with respect to published results. To allow reproducibility of their work, detailed procedures are provided. Implementation results are presented for different configurations of infinite impulse response filters. More, the impact of the proposed bit-width optimisation on the filter poles and zeros is provided to show the effectiveness of the proposed solution. Their solution provides overall improvement going up to 17% of the circuit's area with respect to existing methods. The proposed technique for uniform fractional bits allocation runs in a negligible time independently of the targeted accuracy.

1 Introduction

The reflex for a designer to ensure accuracy is to widen word-lengths leading to the use of prohibitive hardware resources or to elaborated time multiplexing architectures. The obvious and gradual expansion of arithmetic operators when going through pipeline and feedback paths, the need for rapid delivery, for re-usability and for generosity prevent from looking to other ways. Often, uniform and oversized buses are intentionally used even at deep levels of design hierarchies. However, the related cost is not always acceptable. Actually, in the literature, several references [1–6] propose analysis approaches and optimisation techniques to get to the best word-lengths to accuracy tradeoffs. Their efforts are dedicated to circuits using fixed point number representation. For such cases, the bit-width optimisation (BWO) problem consists of minimising the number of bits for the integer part (IB) to prevent overflow and minimise the number of bits for the fractional part (FB) to meet accuracy requirements through range analysis and precision analysis, respectively.

Taking careful look at BWO approaches in the literature, we distinguish two main classes. Dynamic analysis [7–9] which can take a considerable amount of simulation time and does not guarantee underestimation avoidance since it depends on input stimuli of simulations. While static analysis [1–4, 10] considers design architectures and related input characteristics. Consequently, this analysis overestimates bit-widths to avoid violation of accuracy constraints. Our approach belongs to the static analysis class.

Different precision analysis is considered depending on the metric error that they dealt with. For instance, in [11, 12], the considered metric error is the maximum mean-square error and the signal-to-quantisation-noise ratio. The latter are related to the signal-to-noise concept. However, these methods ignore the coefficient quantisation errors what leads to underestimation of the error's bound. In our contribution, all sources of errors are considered. It gives more accuracy to our analysis. Authors in [13–16] considered the maximum mismatch (MM) as a metric error. However, in this paper, we target MM and performed specific bit-width allocation separately for each source of error, which gives more robustness to our method.

Authors in [1, 2, 17–23] propose specified methods for feed-forward circuits. Moreover, authors in [18–23] target BWO of arithmetic circuits specified by polynomials. Obviously, circuits with feedbacks need specific approach to consider the complication added by feedback loops. In other hand, the approaches proposed in [11, 13, 24] calculate the range analysis. They use L^1 norm given by the sum of the absolute values of the circuit outputs. The solutions based on L^1 norm are robust, but provide overestimations of the exact range. Authors in [25, 26] use the affine arithmetic (AA) for the same reason. Interval arithmetic (IA) approach establishes worst case bounds on each intermediate step of the calculation by setting worst case bounds on individual operations. However, since dependencies between intermediate variables are not taken into account, the maximum bounds obtained using IA is much larger than the actual possible maximum bounds of values. AA approach was adapted to deal with interdependencies between variables. Nevertheless, non-affine operations such as multiplications are substituted by affine approximation while inserting new variables. While AA is often much better than IA, it can still give an overestimate of upper bounds. Particularly when strongly non-affine operations occur on the calculation path. Since the feedback part of infinite impulse response (IIR) filter contains multiplications, the use of AA overestimates the upper bounds and therefore does not find the best bit-width. A significant word-length reduction for circuits with feedbacks is presented in [14–16].

As stated before, interconnections could have different lengths. In this paper, we present an interconnection classification according to computation paths so that a maximum value is determined for each segment. The latter gives the integer bits IBs. For fractional bits (FBs), we use initial uniform fractional bit (IUFB) as the computable lower bound for the (UFBs). Then, we analytically compute the FB for the intermediate variables and for the output z_n (Fig. 1). Further improvements were brought by refining the FB of output data z_n and the FB of the input data x_n. In this paper, based on a new output error model, we are giving every detail of that solution and we are proving the efficiency of our proposed solution by presenting the implementation results and

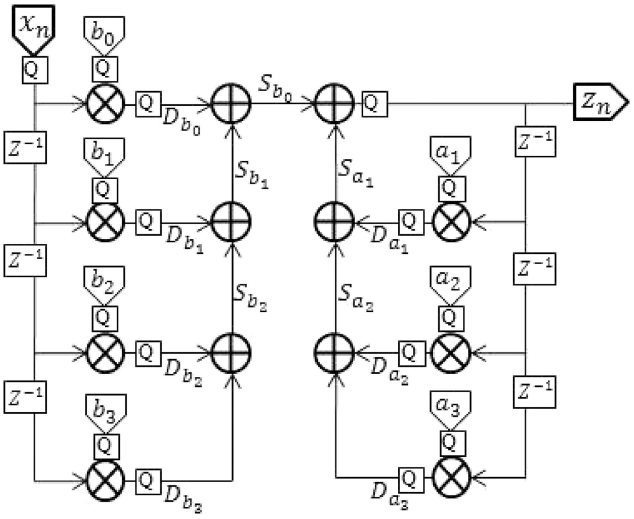

Fig. 1 *Direct form 1 structure of a third-order IIR filter*

analysing the impact of our optimisations on the poles of an IIR filter.

This paper is organised as follows: We introduce the basic concepts and definitions in Section 2. Section 3 is devoted to the way we calculate the IBs of each segment. In Section 4, we present the applied approach to compute maximum imprecision. In Section 5, we compare experiment results to benchmarks as well as a synthesis on related improvements. Section 5.1 shows the impact of our optimisations on the poles of an IIR filter. A conclusion ends the paper.

2 Basic concepts and definitions

Basic concepts and definitions are reviewed briefly in this section.

Linear circuits with feedbacks (LCFs) are defined as circuits where, in polynomial representation, the degree's sum of each monomial is one. In our approach, we adopted IIR filter (1) which is a common example of LCF in order to compare our results with those of similar works

$$z_n = \sum_{i=0}^{P} b_i x_{n-i} + \sum_{j=1}^{Q} a_j z_{n-j}, \tag{1}$$

where P and Q are the order of the feed-forward part and the order of the feedback part, respectively, b_i and a_i are the feed-forward and feedback filter coefficients, respectively.

IIR filter formula (1) can be given as $[z] = [x] * [h]$, where $[h]$ is the impulse response and '*' is the convolution operator. Transfer function provides a basis for determining important system response characteristics. A rational transfer function for a discrete-time LTI system (linear time-invariant) has the form:

$$H[z] = \frac{\sum_{i=0}^{P} b_i z^{-i}}{1 - \sum_{j=1}^{Q} a_j z^{-j}}. \tag{2}$$

2.1 Poles and zeros

The roots of the equation $\sum_{i=0}^{P} b_i z^{-i} = 0$ are defined to be the system zeros denoted by the O_i, and the roots of the equation $1 - \sum_{j=1}^{Q} a_j z^{-j} = 0$ are defined to be the system poles denoted by the p_i.

The transfer function can be rewritten in terms of the poles p_i and the zeros O_i as follows:

$$H[z] = z^{Q-P} b_0 \frac{\prod_{i=1}^{P} (z - O_i)}{\prod_{j=1}^{Q} (z - p_j)}. \tag{3}$$

2.2 Stability system

IIR filter is implementable if the output bound is always finite for an arbitrary bounded input. Means that the IIR filter must satisfy the bounded input bounded output stability conditions. Causal LTI IIR systems are stable if all of the poles of the system function lie inside the unit circle.

The range of the output z_n is computed using the approach described in [14], therefore after n iterations, the output bound $B_o = (B_{\text{olow}}[n], B_{\text{oupp}}[n])$ is defined as

$$B_{\text{oupp}}[n] = \max\{\max(z_n), \max(z_{n-1}), \ldots, \max(z_0)\},$$
$$B_{\text{olow}}[n] = \min\{\min(z_n), \min(z_{n-1}), \ldots, \min(z_0)\},$$

where $\max(z_i)$ and $\min(z_i)$ are the maximum and minimum value of $z[i]$, respectively, for $i = 0, \ldots, n$.

In accordance with the concept of convergence, since B_{oupp} and B_{olow} are monotonically increasing and decreasing functions, respectively, when the number of iterations n approaches to infinity, B_{oupp} and B_{olow} reach the exact upper and lower bounds of the output z_n. However, for a sufficient condition to convergence, a window of a suitable height equal to W is needed. Idem $|B_{\text{oupp/low}}[n] - B_{\text{oupp/low}}[n - W]|$ is a monotonically decreasing function with respect to n and $\lim_{n \to \infty} (B_{\text{oupp/low}}[n] - B_{\text{oupp/low}}[n - W]) = 0$.

2.3 Fixed point number representation

Each fixed-point signal z has an integer bit-width (IB) and an FB-width. Hence, the total width of z is equal to $\text{IB}_z + \text{FB}_z$. IB sets the range, while FB sets the precision. The bit-width allocation problem consists of determining the IBs and the FB*s* through range analysis and precision analysis, respectively.

3 Range analysis

The effectiveness of our solution concerns both IB and FB widths. In this section, we present the IB determination process for each interconnection.

Authors in [14] proposed an efficient method for the output range calculation. We use this method to calculate the range and the IBs of z_n. Once the limit of the output is calculated, we just need to loop it once more to calculate the limits of the intermediate segments of the feedback part. Considering that interconnections are classified and labelled according to their related calculating path (see Fig. 1), we calculate the maximum value for each interconnection as indicated below:

- x_n: the input signal, its maximum value is $\max(|x_{min}|, |x_{max}|)$,
- b_i: the segments of feed-forward coefficients, $i = 0, \ldots, P$,
- a_j: the segments of feedback coefficients, $j = 1, \ldots, Q$.

In the direct form of the IIR filters (Fig. 1), we have three types of intermediary variables.

- The first type is labelled by D_{b_i} and D_{a_i} which represent the multiplication output of the input x_{n-i} by coefficient b_i and the output z_{n-j} by coefficient a_j, respectively, their maximum values are $\max(|b_i x_{min}|, |b_i x_{max}|)$ and $\max(|a_j z_{min}|, |a_j z_{max}|)$, respectively.
- The second type is labelled by S_{b_i} and S_{a_i} which represent the segments between two adders in feed-forward part and in feedback part, respectively, their ranges are $\left[\min_{S_{b_k}}, \max_{S_{b_k}}\right]$ where

$$\min_{S_{b_k}} = \sum_{i=k}^{p} \min(x_{\min} b_i, x_{\max} b_i),$$

$$\max_{S_{b_k}} = \sum_{i=k}^{p} \max(x_{\min} b_i, x_{\max} b_i),$$

and $\left[\min_{S_{a_l}}, \max_{S_{a_l}}\right]$ where

$$\min_{S_{a_l}} = \sum_{j=l}^{Q} \min(z_{\min} a_j, z_{\max} a_j),$$

$$\max_{S_{a_l}} = \sum_{j=l}^{Q} \max(z_{\min} a_j, z_{\max} a_j),$$

for $k = 0, \ldots, P-1$ and $l = 1, \ldots, Q-1$, respectively.

- The third type is the output z_n. It is considered as an intermediary variable because it is an output of adder and an input of multiplications with feedback coefficients a_j (Fig. 1).

The IB number of each segment is given as

$$\mathrm{IB}_{\mathrm{val}_{\max}} = \begin{cases} 1 & \text{for } \mathrm{val}_{\max} < 1, \\ \lfloor \log_2(\mathrm{val}_{\max}) \rfloor + 2 & \text{otherwise}. \end{cases} \quad (4)$$

where ($\mathrm{val}_{\max}$) is the maximum values of each segment.

4 Precision analysis

In this section, the objective is to optimise the FB for a predefined precision. Such optimisation leads to reduction in the circuit area, hence in its cost. In the sequel, we propose a new method to evaluate the computational error. To develop a general error propagation formula, we find its upper limit using the desired bit widths. When this limit reaches the maximum required error (E_{req}), the bit widths considered are taken. The proposed process is a refinement process in several steps (Fig. 2). In the first step, we analytically calculate the IUFB which is the lower bound for the UFB. The latter is computed with an incremental search algorithm using the IUFB values. In the second step, an adjustment algorithm allocates a specific FB value for each coefficient ($\mathrm{FB_c}$). In the next step, an effective FBFB width of the intermediate variables (FB_{iv}) is computed by an analytic formula. The final step refines these results by reducing as much as possible the FB_z and the FB_x of the output and input signal, respectively.

For best accuracy, we modify the maximum error models by considering the quantisation error for each segment.

4.1 Error's propagation model

In this subsection, we present an error's propagation formula for an IIR filter. In [14], the authors represent the fixed-point implementation of z_n in (1) as follows:

$$z_n^{fxd} = \sum_{i=0}^{P} ((b_i - e_{b_i})(x^{-i} - e_{in}) - e_{tr}) + \sum_{j=1}^{Q} ((a_j - e_{a_j})(z_{n-j}^{fxd}) - e_{tr}). \quad (5)$$

Taking into account all the segment widths and, thus, considering the quantisation errors, the fixed-point implementation of (1) is given as

$$z_n^{fxd} = \sum_{i=0}^{P} ((b_i - e_{b_i})(x_{n-i} - e_{x_{n-i}}) - Q(x_{n-i} b_i)) + \sum_{j=1}^{Q} ((a_j - e_{a_j})(z_{n-j} - err_{z_{n-j}}) - Q(z_{n-j} a_j)). \quad (6)$$

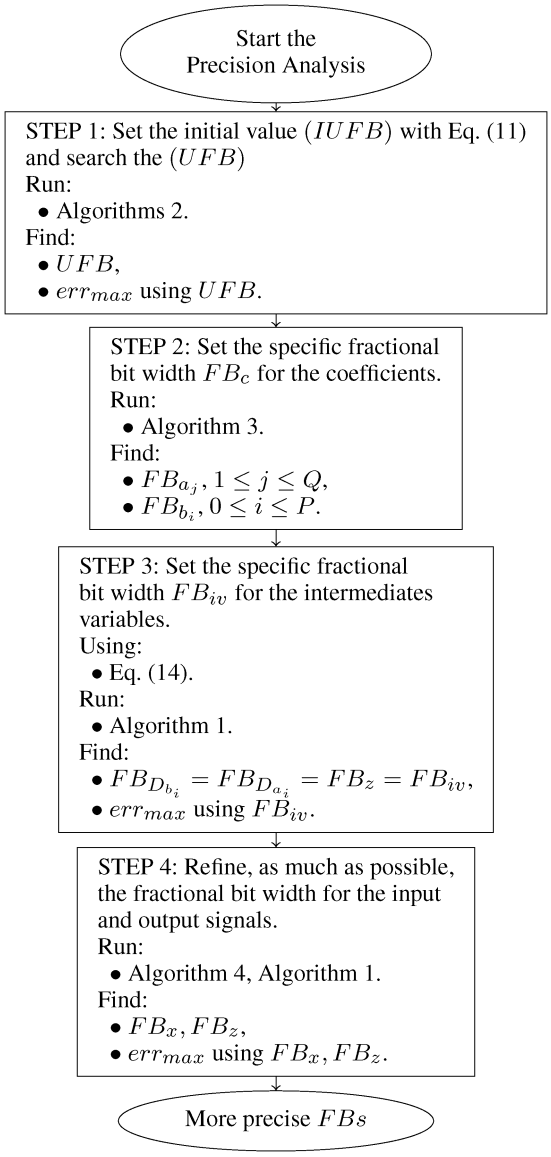

Fig. 2 *FB width optimisation process*

where e_{b_i} and e_{a_j} are the quantisation errors of b_i and a_j coefficients, $Q(x_{n-i} b_i)$ and $Q(z_{n-j} a_j)$ are the generated quantisation errors of the multiplication results of $(b_i x_{n-i})$ and of $(a_j z_{n-j})$, respectively, at the segments labelled D_{b_i} and D_{a_j}, respectively (Fig. 1).

After simplification of (6), we obtain as

$$z_n^{fxd} = z_n - \mathrm{err}_{z_n}, \quad (7)$$

where err_{z_n} can be written as follows:

$$\mathrm{err}_{z_n} = \sum_{i=0}^{\min(P,n)} (e_{b_i} x_{n-i}^{fxd} + e_{x_{n-i}} b_i + Q(x_{n-i} b_i)) + \sum_{j=1}^{\min(Q,n)} (e_{a_j} z_{n-j} + \mathrm{err}_{z_{n-j}} (a_j - e_{a_j}) + Q(z_{n-j} a_j)). \quad (8)$$

After developing the error formula of (8), we deduce that the flattened expression can be written as follows:

$$\mathrm{err}_{z_n} = \sum_{i=0}^{n} (\gamma_i x_{n-i} + C_i^{fxd} e_{x_{n-i}}) + QZ_n, \quad \text{where for } i \in \{0, 1, \ldots, n\}, \quad (9)$$

$$\gamma_i = \sigma_i + \sum_{k=1}^{\min(i,Q)} a_k^{fxd}\gamma_{i-k}, \tag{9a}$$

$$\sigma_i = e_{b_i} + \sum_{k=1}^{\min(i,Q)} e_{a_k} C_{i-k}, \tag{9b}$$

$$C_i = b_i + \sum_{j=1}^{\min(i,Q)} a_j C_{i-j}, \quad \text{where } C_i = 0 \text{ if } i < 0 \tag{9c}$$

$$QZ_n = \sum_{k=0}^{n} A_{n-k}\left(Q(z_k) + \sum_{i=0}^{\min(k,P)} Q(x_{k-i}b_i) + \sum_{j=1}^{\min(k,Q)} Q(z_{k-j}a_j)\right), \tag{9d}$$

where $Q(z_k)$ is the quantisation error of z_k

$$A_n = \sum_{j=1}^{\min(Q,n)} a_j A_{n-j}, \quad \text{where } A_0 = 1, \tag{9e}$$

It is clear that the error calculated by (9) is not always positive. Hence, we need to define an upper bound for (9) as presented in the following subsection.

4.2 Upper bound for the propagation error

In order to find the upper bound of the error, we must find the upper bound of each term in (9), which will be used to determine the most appropriate corresponding FB*s*.

As we have $\forall n \geq 0$

$$err_{z_n} \leq \max\left(|\max_{\gamma x}|, |\min_{\gamma x}|\right) + \max_{QZ_n} + \sum_{i=0}^{n} \max(2^{-\mathrm{FB}_{x}-1} C_i^{fxd}, -2^{-\mathrm{FB}_{x}-1} C_i^{fxd}), \tag{10}$$

where

$$\sum_{i=0}^{n}(\gamma_i x_{n-i}) \leq \sum_{i=0}^{n}\max(\gamma_i x_{\min}, \gamma_i x_{\max}) = \max_{\gamma x}, \tag{10a}$$

$$\sum_{i=0}^{n}(\gamma_i x_{n-i}) \geq \sum_{i=0}^{n}\min(\gamma_i x_{\min}, \gamma_i x_{\max}) = \min_{\gamma x}, \tag{10b}$$

$$\max_{QZ_n} = \sum_{k=0}^{n}|A_{n-k}|\left(2^{-(\mathrm{FB}_z+1)} + \sum_{i=0}^{\min(k,P)} 2^{-(\mathrm{FB}_{D_{b_i}}+1)} + \sum_{j=1}^{\min(k,Q)} 2^{-(\mathrm{FB}_{D_{a_j}}+1)}\right) \geq QZ_n. \tag{10c}$$

This upper bound of the error ((10)) is used in the subsequent subsections for the calculation of the maximum error generated by the FB widths. For example, when looking for the UFB in the flowing subsection, we set the quantisation error of the input signal ($e_{x_{n-i}}$), of the intermediate variables ($Q(x_{k-i}b_i)$) and ($Q(z_{k-j}a_j)$) to $2^{-\mathrm{UFB}-1}$. However, in the general case, each quantification error of an interconnection ($s \in \{x_n, z_n, D_{b_i}, D_{a_j}\}$) is replaced by $2^{-\mathrm{FB}_s-1}$.

4.3 Error estimation and FB allocation

To get an error estimation, we use the upper bound ((10)) of the error formula ((9)). Algorithm 1 (see Fig. 3) illustrates the pseudo code to calculate the error limit for a sequence of widths ($\{\mathrm{FB}_x, \mathrm{FB}_z, \mathrm{FB}_{b_i}, \mathrm{FB}_{D_{b_i}}, \mathrm{FB}_{S_{b_i}}, \mathrm{FB}_{a_j}, \mathrm{FB}_{D_{a_j}}, \mathrm{FB}_{S_{a_j}}\}$) using the upper bound ((10)) and (9a), (9b) and (9c). The parameter ε is a convergence indicator. Its value must be much smaller than E_{req}, e.g. $\varepsilon = E_{\mathrm{req}}/1000$. For a given sequence of widths, the loop at line 7 of Algorithm 1 ends either when the process converges or when $\mathrm{err}_{z_n} > E_{\mathrm{req}}$.

Input: FBs, E_{req} , b , a , $[x_{min}, x_{max}]$.
Output: err_{max} : maximum imprecision.

1 **begin**
2 $Set\ W\ to\ an\ apropriate\ value$
3 $\varepsilon = E_{req}/1000$
4 $err_{max} = 0;\ n = 0;$
5 $a_j^{fxd}(j = 1..Q)$: The values of a_j after quantification;
6 $b_i^{fxd}(i = 0..P)$: The values of b_i after quantification;
7 **while** (converge=0) **do**
8 Calculate C_n by Eq. (9c)
9 Calculate C_n^{fxd} by Eq. (9c)
10 Calculate σ_n by Eq. (9b)
11 Calculate γ_n by Eq. (9a)
12 Calculate A_n by Eq. (9e)
13 Calculate $max_{\gamma x}$ by Eq. (10a)
14 Calculate $min_{\gamma x}$ by Eq. (10b)
15 Calculate max_{QZ_n} by Eq. (10c)
16 Calculate err_{z_n} by Eq. (10)
17 $B_{e_{upp}}[n] = max(B_{e_{upp}}[n-1], err_{z_n})$
18 $err_{max} = B_{e_{upp}}[n]$
19 **if** $(err_{z_n} < E_{req})$ **then**
20 **if** $(n > W)$ **then**
21 $\Delta[n, W] = B_{e_{upp}}[n] - B_{e_{upp}}[n-W]$
22 **if** $\Delta[n, W] \leq e$ **then**
23 $Converge = 1$
24 $n++$
25 **else**
26 $Converge = -1;$
27 **return** err_{max}

Fig. 3 *Algorithm 1: Calculate the error limit for a sequence of widths*

4.3.1 UFB allocation: In large-scale systems, runtime is a trivial parameter of the optimisation process. In the case where the value of UFB is really large, runtime increases with respect to the number of reliability tests. Therefore, we present in this section an IUFB which represents the initial value used for the UFB search. IUFB is analytically computed using

$$\mathrm{IUFB} = \left\lfloor -\log_2\left(\frac{E_{\mathrm{req}}}{\max_{|x|}(P+1) + \max_{|z|}Q}\right) - 1 \right\rfloor, \tag{11}$$

where $\lfloor a \rfloor$ is the floor function, which gives the greatest integer $\leq a$.

Proof: We have if UFB is realisable then (12) is verified

$$\sum_{i=0}^{P} e_{b_i} x_{n-i} + \sum_{j=1}^{Q} e_{a_j} z_{n-j} \leq E_{\mathrm{req}}. \tag{12}$$

The aforementioned (12) considers the product of the quantification error of b_i and a_j coefficients in one hand, and the maximum values of the input signal x_n and the output signal z_n, respectively. These latter, must be less than the required error (E_{req}).

Using UFB, the quantification error of coefficients e_{b_i} and e_{a_j} is bounded by $2^{-\mathrm{UFB}-1}$. Then (12) becomes

$$2^{-\mathrm{UFB}-1} \leq \frac{E_{\mathrm{req}}}{(P+1)\max_{|x|} + Q\ \max_{|z|}}. \tag{13}$$

Then

$$\mathrm{UFB} \geq -\log_2\left(\frac{E_{\mathrm{req}}}{(P+1)\max_{|x|} + Q\ \max_{|x|}}\right) - 1$$

□

Algorithm 2 (see Fig. 4) shows our iterative search process that initialise UFB at IUFB and increments it if $\text{err}_{\max} > E_{\text{req}}$.

4.3.2 Coefficients fractional bits width (FB_c) *allocation:* In this subsection, we calculate the specific FB width FB_c for all coefficients starting from UFB.

When the least significant bits of an FB are zeros, one can remove them without affecting the related error. Then, for a such case, smaller FB width can be obtained for the exact same coefficient value.

Our proposed Algorithm 3 (see Fig. 5) returns the specific fractional width for each coefficient.

4.3.3 Fractional bits width allocation for intermediate variables (FB_{iv})*:* In the direct form of the IIR filters (Fig. 1), we have three types of intermediate variables. Their respective FB_s is calculated as follows:

Using preliminary simplification method proposed in [17], we re-attribute the values of $\text{FB}_{S_{b_i}}$ and $\text{FB}_{S_{a_j}}$ for the S_{b_i} and S_{a_j} interconnections, respectively. As S_{b_i} and S_{a_j} are between two additions, we set $\text{FB}_{S_{b_i}}$ and $\text{FB}_{S_{a_j}}$ to the maximum FB of input signals. Thus, quantisation errors of these interconnections are reduced to zero.

We assign the same FBs width FB_{iv} for the variables labelled by D_{b_i}, D_{a_i} and the variable of output z_n. FB_{iv} is analytically calculated starting from UFB as

$$\text{FB}_{iv} = \left\lfloor -\log 2\left(\frac{\Delta\text{err}}{\sum_{i=0}^{l}|A_i|(P+Q+1)}\right) - 1 \right\rfloor \tag{14}$$

where Δerr is the difference between the required error E_{req} and the maximum error generated using UFB ($\text{err}_{\max}$) without considering the intermediate quantisation errors $(P+Q+1/2^{\text{UFB}+1})\sum_{i=0}^{l}|A_i|$. The Δerr is calculated as

$$\Delta\text{err} = E_{\text{req}} - \text{err}_{\max} + \frac{P+Q+1}{2^{\text{UFB}+1}}\sum_{i=0}^{l}|A_i| \,. \tag{15}$$

4.3.4 Output and input FB refinement 'OIFWR': Since the input and the output signals are involved in all the multiplications at the feed-forward as well as at the feedback parts. More, the value of $\text{err}_{\max}$ is monotonically decreasing as well as the values of FB_z and FB_x are increasing. So, we propose a refinement procedure for the output z_n and input x_n, the related pseudo code is shown in Algorithm 4 (see Fig. 6). In this step, we decreased the FB_z and FB_x by one as long as $\text{err}_{\max} < E_{\text{req}}$.

5 Experimental results

We present the results of our approach and we compare them with recent work to show its efficiency. All the used algorithms, ours and the benchmarks, have been implemented with MATLAB and with a Cadence incisive verification tool. For the ASIC implementation, we used the synthesis flow using *Cadence RTL compiler*. The area reports for every studied case are obtained after synthesis running using a Taiwan Semiconductor Manufacturing Company Complementary Metal Oxide Semiconductor 0.18 μm Process Design Kit.

The benchmarks we used are summarised in Table 1. These benchmarks were used by recent work such as [14–16].

Equation (11) gives the best results for IUFB for all values of E_{req} we analysed $(10^{-1}, 10^{-2}, \ldots, 10^{-10})$ and for all of benchmarks (Table 1). As shown in Fig. 7, the different results of IUFB and UFB for the four benchmarks are (B_0, B_1, B_2, B_3). For example, in sub-figure B_1 of Fig. 7, the values of UFB are equal to IUFB + 1 for all analysed E_{req} values except for $E_{\text{req}} = 10^{-6}$ where UFB = IUFB + 2.

For the run time, as already mentioned in Section 4.3.1 and as shown in Fig. 8, the black part of each bar represents the time spent

Input: E_{req} , P , Q , $Max_{|x|}$, $Max_{|z|}$.
Output: err_{max}, UFB.
begin
  Calculate $IUFB$ with Eq. (11)
  $UFB = IUFB$
  Calculate err_{max} with Algorithm 1
  while $(err_{max} > E_{req})$ **do**
    $UFB = UFB + 1$
    Calculat err_{max} with Algorithm 1
  return UFB

Fig. 4 *Algorithm 2: Calculate IUFB and search UFB*

Input: UFB , a_j , b_i.
Output: FB_{a_j},FB_{b_i}.
begin
  for $j := 1$ to Q **do**
    $FB_{a_j} := UFB$
    Calculate $a_j(FB_{a_j})$: the value of a_j with FB_{a_j} bits fractional
    while $(a_j(FB_{a_j}) = a_j(FB_{a_j} - 1))$ **do**
      $FB_{a_j} := FB_{a_j} - 1$
  for $i := 0$ to P **do**
    $FB_{b_i} := UFB$
    Calculate $b_i(FB_{b_i})$: the value of b_i with FB_{b_i} bits fractional
    while $b_i(FB_{b_i}) = b_i(FB_{b_i} - 1)$ **do**
      $FB_{b_i} := FB_{b_i} - 1$
  return FB_{a_j}, FB_{b_i}

Fig. 5 *Algorithm 3: Search FB_c*

Input: E_{req}, FB_x, FB_z.
Output: FB_x, FB_z.
begin
  while $(err_{max}() < E_{req})$ **do**
    $FB_z := FB_z - 1$;
    $FB_x := FB_x - 1$;
  $FB_z := FB_z + 1$;
  $FB_x := FB_x + 1$;
  while $(err_{max}() < E_{req})$ **do**
    $FB_z := FB_z - 1$;
  $FB_z := FB_z + 1$;
  while $(err_{max}() < E_{req})$ **do**
    $FB_x := FB_x - 1$;
  $FB_x := FB_x + 1$;
  return FB_x, FB_z;

Fig. 6 *Algorithm 4: Output and input fractional widths refinement (OIFWR)*

Table 1 Benchmarks

Bench	Order	Nominator full-precision coefficients	Denominator full-precision coefficients
B_0	2	101.8, −203.4, 101.6	−1.967, 0.968
B_1	4	0.03752, 0.150086, 0.22513, 0.150086, 0.03752	−1.1839, 1.366039, −0.7782356, 0.2671877
B_2	4	0.570925344, −2.280504, 3.41916, −2.2805, 0.5709253442	−3.07156, 3.68147, −2.03462, 0.4474244
B_3	6	−1.5608, 3.07, 3.07, −1.5608, 1	−2.547, 4.2203, −4.3179, 3.0547, −1.3498, 0.3168

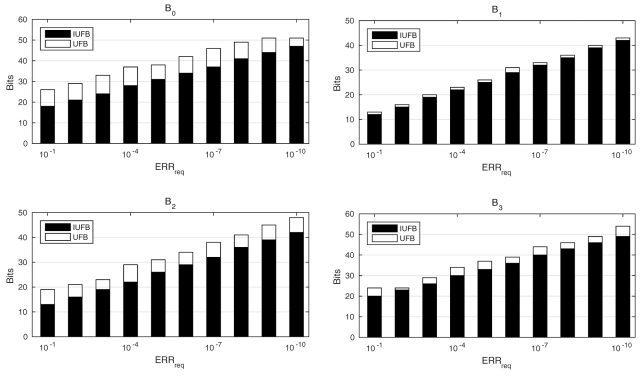

Fig. 7 *IUFB and UFB estimation*

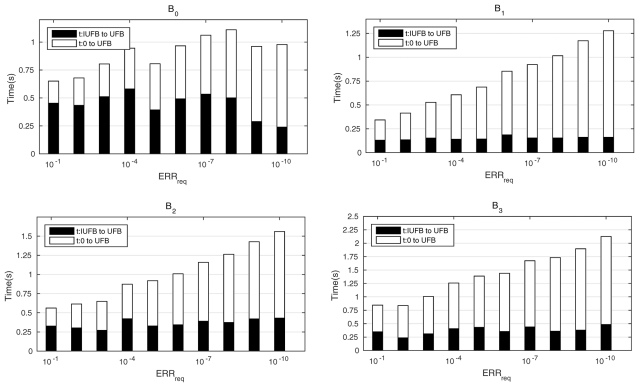

Fig. 8 *Time estimation to calculate UFB*

Table 2 Comparison of our algorithm to existing methods when $E = 0.1$

	Step 1		Step 2	Step 3	Step 4
	IUFB	UFB	FB_c	FB_{iv}	IOFWR
	(11)	Alg. 2	Alg. 3	(14)	Alg. 4
	20	24	$FB_a = 24, 24, 24, 23, 23, 23$	$FB_{D_{b_i}} = 15$	
				$FB_{D_{a_j}} = 15$	$FB_x = 12$
			$FB_b = 24, 24, 24, 24, 0$	$FB_z = 15$	$FB_z = 11$
error	>0.1	0.0093	0.0093	0.022	0.087
area	\	587,912	543,624	426,494	344,322

Table 3 Experimental results: evaluating the proposed algorithms on several IIR filter benchmarks

Bench	FBs allocation		$\text{err}_{\max}$ ($E_{\text{req}} = 0.01$)		Area (percentage gain)		Dominant poles in the Z plane	
	[15]	This paper	[15]	This paper	[15]	This paper	reference	quantised
B_0	$FB_z = 21$	$FB_z = 17$	0.008	0.009	255,660	243,689	0.9835	0.9835
	$FB_a = 29$	$FB_a = 27, 29$					$\pm 0.02697684i$	$\pm 0.02697688i$
	$FB_b = 29$	$FB_b = 28, 29, 29$						
	$FB_S = 21$	$FB_{iv} = 21$				(4.68%)		
	$FB_x = 15$	$FB_x = 17$						
B_1	$FB_z = 14$	$FB_z = 11$	0.0067	0.0097	169,812	147,592	0.14354	0.14353
	$FB_a = 16$	$FB_a = 14, 14, 15, 15$				(13.09%)	$\pm 0.85662i$	$\pm 0.85664i$
	$FB_b = 16$	$FB_b = 15, 14, 15, 14, 15$						
	$FB_S = 14$	$FB_{iv} = 16$						
	$FB_x = 11$	$FB_x = 11$						
B_2	$FB_z = 19$	$FB_z = 15$	0.0083	0.0098	249,528	240,190	0.9118	0.9118
	$FB_a = 21$	$FB_a = 18, 20, 21, 21$					$\pm 0.295350i$	$\pm 0.295353i$
	$FB_b = 21$	$FB_b = 21, 19, 20, 21, 21$						
	$FB_S = 19$	$FB_{iv} = 20$				(3.74%)		
	$FB_x = 10$	$FB_x = 13$						
B_3	$FB_z = 17$	$FB_z = 15$						
	$FB_a = 24$	$FB_a = 24, 24, 24, 23, 23, 23$	0.0093	0.0096	455,835	402,163	0.299929056	0.299929059
	$FB_b = 24$	$FB_b = 24, 24, 24, 0$					$\pm 0.92123715i$	$\pm 0.92123716i$
	$FB_S = 17$	$FB_{iv} = 19$				(11.77%)		
	$FB_x = 17$	$FB_x = 16$						

to calculate UFB starting from IUFB. The over-all bar represents the time spent to calculate the same value of the UFB starting from 0. The time spent to calculate UFB when it is initialised to 0, becomes larger if the E_{req} is smaller. In the same figure, if UFB is initialised to IUFB the estimated time becomes more stable and independent from the value of E_{req}. The reason is that the IUFB calculation is based on E_{req}. This allows that IUFB converge to UFB every time when the E_{req} becomes smaller.

Considering the error expressed in (10) and the area of ASIC implementation, we got from the synthesis flow using *Cadence RTL compiler*. Tables 2 and 3 show the improvements in the allocation of different bit widths using our approach compared to [15].

In Table 2, we deal with the bench 3, where we apply the process presented in Fig. 2. Note that the input variables are set to:

- Error required: $E_{\text{req}} = 0.1$,
- Input signal: $x \in [-100, 100]$.

As shown in Step 1 (Table 2), the IUFB is equal to 20, whereas the minimum UFB achievable is equal to 24. Using this value, $\text{err}_{\max}$ is equal to 0.0093 and the circuit requires an area of

Table 4 Shifts between the reference and the quantised poles

Bench	Poles	
	Reference	Quantised
B_0	$0.9835 \pm 0.026976841920432i$	$0.983500000002096 \pm 0.026976841858062i$
B_1	$0.143544682862220 \pm 0.856620231233786i$	$0.143544680661795 \pm 0.856620204396694i$
	$0.448405317137781 \pm 0.391284524171890i$	$0.448405259065988 \pm 0.391284441451659i$
B_2	$0.911751177345619 \pm 0.295344939598673i$	$0.911751162099362 \pm 0.295344938665894i$
	$0.624028822654381 \pm 0.312575000196740i$	$0.624028837469699 \pm 0.312574989224717i$
B_3	$0.299929055932414 \pm 0.921237148665833i$	$0.299929055955453 \pm 0.921237148658756i$
	$0.395086746927798 \pm 0.769733693840343i$	$0.395086746946020 \pm 0.769733693859214i$
	$0.578484197139792 \pm 0.340914357303868i$	$0.578484197108772 \pm 0.340914357257890i$

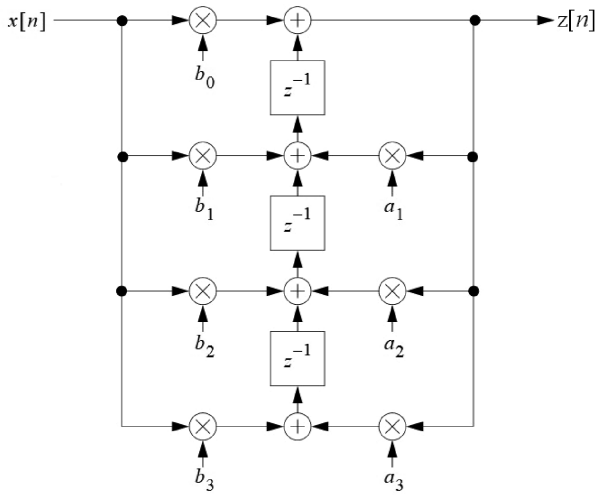

Fig. 9 *Direct form 2 structure of a third-order IIR filter*

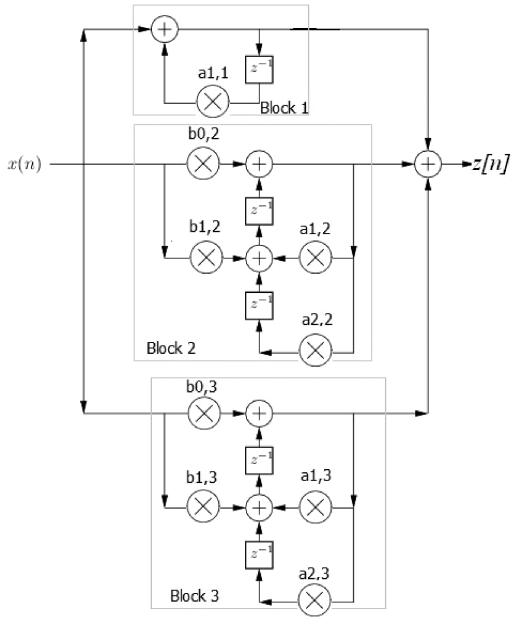

Fig. 10 *Parallel form structure for IIR filter*

587912 μm. In Step 2 (Table 2), we have reduced the FB_c of the coefficients and thus the area becomes 543, 624 μm with no effect on the error. We continue to reduce the area to 426, 494 μm by reducing the fractional intermediate bits and FB_z to 15 and calculating the FB_{iv} by (14). Finally, reducing the FB_x and again the FB_z to 12 and 11, respectively, we achieve an area of 344, 322 μm. Our total area improvement in this table is 41.43% compared to the results in [14] and is 17.36% compared to the results in [15].

The input variables and parameters of all the benchmarks in Table 3 are set to:

- Error required: $E_{\text{req}} = 0.01$,
- Input signal: $x \in [0, 100]$.

Table 3 shows the FBs and the err_{max} obtained in this paper compared to those of [15]. It shows clearly that our method deals effectively with the trade-off between the area and the err_{max} and guarantees circuit's area reduction while respecting the predefined precision constraint ($E_{\text{req}} = 0.01$). In instance for B_1, with respect to [15], we notice a major improvement going up to 13.09% of the circuit's area. We generate also an err_{max} equal to 0.0097 versus 0.0063 generated in [15]. The same improvements are noticed for B_0, B_2 and B_3 with varied proportions. This means that our analysis exploits the error margin more effectively than [15]. In addition, Table 3 gives the exact and the quantised positions for the dominant poles which are inside the unit circle.

5.1 Impact of the proposed BWO on the filter's poles

Note that the FB_s of coefficients a_i is wider, which maintains the position of the poles in the unit circle. Our method not only keeps the poles inside the unit circle but also preserves the position of the poles to keep the filter characteristics.

The efficiency of our proposed method is illustrated in Table 3. The positions of quantised poles are almost the same as the positions of the reference poles. In fact, the gap that occurred after the optimisation is so small that it can be ignored. The last two columns of Table 3 provide a numerical quantification of the gaps for the dominant poles. For example, for the bench 0, there is a gap of $\pm 4 \times 10^{-8}$ on the imaginary component while the gap on the real component is zero.

In Table 4, we present the exact values of the poles in Column 2 and the quantisation values that occurred after the optimisation in Column 3. We considered for this table that the maximum error bound is 0.001 and the input signal is $x \in [-1000, 1000]$. Our results show that the gap is at most equal to $10^{-3}E_{\text{req}}$. For example, in Table 3, $E_{\text{req}} = 0.1$ and the gap is at the worst case $<10^{-4}$. In Table 4 at the worst case, the gap is $<10^{-7}$.

5.2 Coding limitations

We can apply this method directly to different circuit structures such as Figs. 9 and 10. In the case of Fig. 10, each block is treated separately. However, we can adapt our method for other circuit structures by recalculating the error's propagation formula according to the structures and to the points of quantification.

6 Conclusion

This paper presents a novel BWO process for LCF handling fixed point numbers.

Regarding the number's range, for each interconnection we determine a maximum value and we propose a classification according to the related path calculation. We also use sequential and analytical methods to get to the best possible accuracy depending on the case.

The proposed method reduces the used component sizes, and thus it reduces the global area, by refining the data input and the

data output fraction while preserving the output error. We proposed an error expression that takes into account every interconnection's error.

For illustration and to check the effectiveness of our proposed optimisation on the utilised IIR filter, we analyse its impact on the poles position. Our analysis and simulation results maintain the poles inside the unit circle while moving with a very slight lag.

The proposed procedures are described in detail. They are very easily reproducible. The results of our different hardware implementations are provided and compared to benchmarks. They show circuit's area reduction going up to 17% with respect to existing methods.

We are engaged to port our proposed process to more elaborated and larger systems.

7 References

[1] Pang, Y., Radecka, K., Zilic, Z.: 'An efficient hybrid engine to perform range analysis and allocate integer bit-widths for arithmetic circuits'. Proc. ASP-DAC, Yokohama, Japan, 2011, pp. 455–460

[2] Lee, D., Gaffar, A.A., Cheung, R.C.C., *et al.*: 'Accuracy-guaranteed bit-width optimization', *IEEE Trans. Comput-Aided Design*, 2006, **25**, pp. 1990–2000

[3] Kinsman, A.B., Nicolici, N.: 'Bit-width allocation for hardware accelerators for scientific computing using SAT-modulo theory', *IEEE Trans. Comput.-Aided Design*, 2010, **29**, pp. 405–413

[4] Radecka, K., Zilic, Z.: 'Using arithmetic transform for verification of datapath circuits via error modeling'. Proc. IEEE VLSI Test Symp, May 2000, pp. 271–277

[5] Lee, D., Gaffar, A.A., Mencer, O., *et al.*: '*Minibit: bit-width optimization via affine arithmetic*' (DAC, Anaheim, CA, USA, 2005), pp. 837–840

[6] Osborne, W.G., Cheung, R.C.C., Coutinho, J.G.F., *et al.*: 'Automatic accuracy-guaranteed bit-width optimization for fixed and floating-point systems'. Proc. Int. Conf. Field Programmable Logic and Applications (FPL), Amsterdam, Netherlands, 2007, pp. 617–620

[7] Shi, C., Brodersen, R.: 'Automated fixed-point data-type optimization tool for signal processing and communication systems'. Proc. Des. Autom. Conf, San Diego, CA, USA, 2004, pp. 478–483

[8] Kum, K., Sung, W.: 'Combined word-length optimization and high level synthesis of digital signal processing systems', *IEEE Trans. Comput. Aided Design*, 2001, **20**, pp. 921–930

[9] Gaffar, A., Mencer, O., Luk, W., *et al.*: 'Unifying bit-width optimization for fixed-point and floating-point designs'. Proc. IEEE Symp. Field-Programmable Custom Computing, Napa, CA, USA, March 2004, pp. 79–88

[10] Radecka, K., Zilic, Z.: 'Arithmetic transforms of imprecise datapaths by Taylor series conversion'. Proc. Int. Conf. Electron. Circuits System, Nice, France, 2006, pp. 696–699

[11] Oppenheim, A.V., Weinstein, C.J.: 'Effects of finite register length in digital filtering and the fast Fourier transform'. *Proc. IEEE*, 1972, **60**, pp. 957–976

[12] Menard, D., Rocher, R., Sentieys, O.: 'Analytical fixed-point accuracy evaluation in linear time-invariant systems', *IEEE Trans. Circuits Syst. I Regul. Pap.*, 2008, **55**, pp. 3197–3208

[13] Carletta, J., Veillette, R., Krach, F., *et al.*: 'Determining appropriate precisions for signals in fixed-point IIR filters'. Proc. Design Automation Conf., June 2003, pp. 656–661

[14] Sarbishei, O., Pang, Y., Radecka, K.: 'Analysis of range and precision for fixed-point linear arithmetic circuits with feedbacks'. Proc. IEEE HLDVT, Anaheim, FL, USA, June 2010, pp. 25–32

[15] Sarbishei, O., Radecka, K., Zilic, Z.: 'Analytical optimization of bit-widths in fixed-point LTI systems', *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 2012, **31**, pp. 343–355

[16] Lamini, E., Bellal, R., Tagzout, S., *et al.*: 'Enhanced bit-width optimization for linear circuits with feedbacks'. Design and Test Symp. (IDT), Algiers, Algeria, December 2014, pp. 168–173

[17] Vakili, S., Langlois, J.M.P., Bois, G.: 'Enhanced precision analysis for accuracy-aware bit-width optimization using affine arithmetic', *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 2013, **32**, pp. 1853–1865

[18] Chichyang, C.: 'High-order Taylor series approximation for efficient computation of elementary functions', *IET Comput. Digit. Tech.*, 2015, **9**, pp. 328–335

[19] De Caro, D., Napoli, E., Esposito, D., *et al.*: 'Minimizing coefficients wordlength for piecewise-polynomial hardware function evaluation with exact or faithful rounding', *IEEE Trans. Circuits Syst. I Regul. Pap.*, 2017, **64**, pp. 1187–1200

[20] Mahdieh, G., Alizadeh, B., Forouzandeh, B.: 'Improved range analysis in fixed-point polynomial data-path', *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 2017, **36**, pp. 1925–1929

[21] Pang, Y., Yan, Y., Lin, J., *et al.*: 'ICAT: engine to perform range analysis and allocate bit-widths for arithmetic datapaths', *J. Circuits Syst. Comput.*, 2015, **24**, p. 1550020

[22] Pang, Y., Yan, Y., Lin, J., *et al.*: 'Designing optimized imprecise fixed-point arithmetic circuits specified by polynomials with various constraints', *J. −Circuits Syst. Comput.*, 2014, **23**, p. 1450010

[23] Pang, Y., Radecka, K., Zilic, Z.: 'Optimization of imprecise circuits represented by Taylor series and real-valued polynomials', *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 2010, **29**, pp. 1177–1190

[24] Jackson, L.B.: 'On the interaction of round-off noise and dynamic range in digital filters', *Bell Syst. Tech. J.*, 1970, **49**, pp. 159–184

[25] López, J.A., Caffarena, G., Carreras, C., *et al.*: 'Fast and accurate computation of the round-off noise of linear time-invariant systems', *IET Circuits Dev. Syst.*, 2008, **02**, pp. 393–408

[26] Caffarena, G., Carreras, C., Lopez, J., *et al.*: 'SQNR estimation of fixed-point DSP algorithms', *Int. J. Adv. Signal Process.*, 2010, **10**, pp. 1–11