

A First Experimental Study of Fixed-Point Approximate Arithmetic in Recursive Lattice Filters

Koch, Peter; Le Moullec, Yannick

Published in:

2023 IEEE Nordic Circuits and Systems Conference, NorCAS 2023 - Proceedings

DOI (link to publication from Publisher):

[10.1109/NorCAS58970.2023.10305450](https://doi.org/10.1109/NorCAS58970.2023.10305450)

Creative Commons License

CC BY 4.0

Publication date:

2023

Document Version

Accepted author manuscript, peer reviewed version

[Link to publication from Aalborg University](#)

Citation for published version (APA):

Koch, P., & Le Moullec, Y. (2023). A First Experimental Study of Fixed-Point Approximate Arithmetic in Recursive Lattice Filters. In J. Nurmi, P. Ellervee, P. Koch, F. Moradi, & M. Shen (Eds.), *2023 IEEE Nordic Circuits and Systems Conference, NorCAS 2023 - Proceedings* Article 10305450 IEEE (Institute of Electrical and Electronics Engineers). <https://doi.org/10.1109/NorCAS58970.2023.10305450>

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal -

Take down policy

If you believe that this document breaches copyright please contact us at vbn@aub.aau.dk providing details, and we will remove access to the work immediately and investigate your claim.

A First Experimental Study of Fixed-Point Approximate Arithmetic in Recursive Lattice Filters

Peter Koch

Department of Electronic Systems
Aalborg University
Aalborg, Denmark
Email: pk@es.aau.dk

Yannick Le Moullec

Thomas Johann Seebeck Department of Electronics
Tallinn University of Technology
Tallinn, Estonia
Email: yannick.lemoullec@taltech.ee

Abstract—In some situations, the numerical properties of Direct Form I (DF-I) and Direct Form II (DF-II) Infinite Impulse Response (IIR) filter structures may degrade, e.g., when the filter approaches its stability limit. Other filter structures may be numerically more robust under such conditions. For example, an N^{th} order Lattice filter composed of N structurally identical cascaded feed-forward sections can be extended and made recursive, i.e., Lattice IIR filter, thereby implementing zeros as well as poles. Such filters, however, have a significantly higher computational complexity compared to their DF-I and DF-II counterparts, increasing the execution time and the energy consumed per sample period. Approximate Computing (AxC) applied in Lattice IIR filters has not been researched in the scientific literature to date, and therefore we suggest substituting the exact fixed-point multiplications and additions with AxC arithmetic building blocks to potentially reduce the resource overhead. In this first study, we analyze the numerical consequences of using such arithmetic units in the 2^{nd} order recursive Lattice structure and compare its performance against the ordinary DF-II structure. Our findings clearly indicate that under certain conditions AxC arithmetic is a useful approach in recursive Lattice filters.

Index Terms—Approximate Multiplication and Addition, IIR Tapped Lattice and DF-II Filter Structures, Numerical Analysis.

I. INTRODUCTION

Digital filters are used intensively in almost any signal processing application related to wireless communication, speech processing, hearing aids, and many more. For such applications, the ever-increasing requirement for easy, reliable, and long-term portability away from power sockets and heavy power banks challenges the system designer with non-trivial requirements for low power consumption while the physical area and the execution time are metrics which at the same time should also comply with tough specifications. These metrics counteract each other, and thus significantly increase the overall complexity of the design- and exploration trajectory. Finding a reasonable trade-off between such design parameters therefore typically requires a compromise, [1]. In applications which involve human interaction, the system designer may benefit from the fact that the computational accuracy can be somewhat relaxed due to the limited human ability to conduct exact perception, e.g., in a time-varying scenario such as audio and video streaming, it is not possible for humans to capture tiny details, errors, and misalignment, [2]. Taking

advantage of this fact motivates the idea of using arithmetic building blocks made from down-scaled circuitry schemes, which compromises the arithmetic accuracy, but at the same time, and as an intended consequence, also reduces the overall power-, area-, and time consumption. Such circuits belong to the class of AxC arithmetic units, [3].

A. Related work

In recent years, many AxC circuits have been proposed, mainly for addition and multiplication, but circuits for other arithmetic operations also exist, see the survey in [4]. Several studies have demonstrated the applicability of AxC circuits in signal processing algorithms having feed-forward branches only, e.g., Finite Impulse Response filters, the Fast Fourier Transform, and the Discrete Cosine Transform, [5], [6], [7]. For such linear time-invariant algorithms, the inaccuracy, i.e., the numerical noise induced by the AxC circuits, is transferred via certain gain factors directly to the system output. For algorithms characterized by one or more feedback loops, i.e., recursive algorithms, the situation is markedly more tortuous with the risk of enforcing instability, potentially prohibiting the use of AxC circuits in such systems.

This is clearly consolidated by the fact that very few available works report on AxC arithmetic being used in recursive DSP algorithms, the exception being [8] which discusses the design and implementation of a 6^{th} order digital A-weighting filter consisting of a cascade of 1^{st} order IIR DF-I sections. The work illustrates that AxC multiplication dictates how the filter sections should be sequentially ordered inside the cascade, and further that the filter amplitude response becomes a function of the degree of approximation introduced in the multipliers. However, this exploratory design reports only on 1^{st} order DF-I structures which is a severe limitation due to the generally accepted design approach where 2^{nd} order filter sections are used for implementing higher-order filters. This issue is addressed in [9] where the authors experiment with AxC multiplication in different types of bi-quad filter structures. For varying pole location and different filter topologies (DF-I, DF-II, and the Direct Canonical Form), they investigate how the degree of inaccuracy in AxC multiplication impact the overall numerical performance of such structures and show that, for a high degree of approximation and for critical pole locations,

the DF-II has, with some exceptions, the best performance, i.e., the least numerical deviation on the output as compared to an equivalent floating-point implementation.

B. Contribution

Both of the above works, however, consider AxC in terms of multiplication only, and similarly, they are limited to the traditional Direct Form filter structures. In order to extend these previous works and broaden the scope of AxC as applied to recursive filter structures, we investigate the impact¹ of AxC multiplication as well as AxC addition on the performance of *i*) the DF-II structure, and *ii*) a recursive Lattice structure known as the IIR Tapped Lattice Structure (ITLS). This recursive structure is, similarly to the traditional feed-forward Lattice structure, known for its superior performance when executed in a fixed-point environment, [10]. To our knowledge, no studies have previously investigated a combined effort to use AxC multiplication and AxC addition in recursive filters, neither the Direct Form filter, nor the Lattice filter.

II. THE IIR TAPPED LATTICE STRUCTURE

Digital filters are defined by their transfer function

$$H(z) = \frac{\sum_{i=0}^M b_i z^{-i}}{1 - \sum_{j=1}^N a_j z^{-j}} = \frac{B(z)}{A(z)} \quad (1)$$

which prior to fixed-point implementation is normally rewritten into either a cascaded or a parallel form of 2^{nd} order sections using factorization or partial fraction expansion, respectively. Many DSP designers prefer the cascaded form, which for $N = M$ leads to

$$H(z) = \prod_{p=1}^{N/2} H_{2,p}(z) = \prod_{p=1}^{N/2} \frac{B_{2,p}(z)}{A_{2,p}(z)} = \prod_{p=1}^{N/2} \frac{\sum_{i=0}^2 b_{i,p} z^{-i}}{1 - \sum_{j=1}^2 a_{j,p} z^{-j}} \quad (2)$$

In this work we investigate exclusively the fundamental 2^{nd} order section. For recursive filters, it is well-known that the pole locations have a notable impact on the numerical properties of a fixed-point implementation. The zero locations, on the contrary, do not affect these properties to the same extent, [11]. Therefore, we experiment with a fixed location of the zero-pair, and a variable location of the pole-pair, both pairs being complex conjugated in the z -plane. For the 2^{nd} order transfer function $H_2(z)$, we therefore choose the polynomials $B_2(z)$ and $A_2(z)$ such that

$$H_2(z) = \frac{1 - \frac{1}{\sqrt{2}} z^{-1} + z^{-2}}{1 + 2r \cdot \cos(\theta_\ell) z^{-1} - r^2 z^{-2}} \quad (3)$$

where the zeros are located on the unit circle at $\omega_{zero} = \pm 1.209391\dots$ radians, and the poles are located at $z = r \cdot e^{\pm j\theta_\ell}$. We opt for $r \in \{0.8, 0.99\}$ and $\theta_\ell = \frac{\ell\pi}{16}$, $\ell = 1..15$.

¹Our work is relevant to a wide range of signal processing applications as mentioned above. Therefore, in this first study we are mainly interested in investigating the numerical implications of using AxC, thus leaving results on specific hardware-related metrics for future research.

In the time-domain, Equ. (3) is expressed by the linear time-invariant difference equation

$$y[n] = \sum_{j=1}^2 a_j \cdot y[n-j] + \sum_{i=0}^2 b_i \cdot x[n-i] \quad (4)$$

which can be real-time executed using various implementation structures. We experiment with the ordinary DF-II structure and the more sophisticated ITLS structure, Fig. 1. The ITLS consists of a recursive part characterized by the k -coefficients (known as the reflection coefficients), and a non-recursive part, i.e., the tapped section, defined by the α -coefficients.

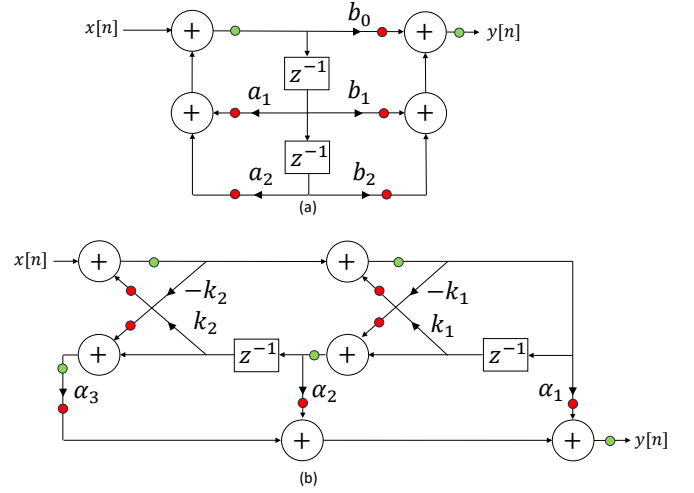


Fig. 1. The two 2^{nd} order filter structures: (a) DF-II, and (b) ITLS. These filters have identical floating-point I/O-relation $H_2(z)$. For a minimum-phase $A_2(z)$ polynomial, the condition $|k_i| < 1$ enables improved numerical robustness of the ITLS structure as compared to the DF-II structure.

It is beyond the scope of this paper to discuss in detail the procedure for converting the (a, b) coefficients into the (k, α) coefficients. In short, however, the Gray-Markel Method, [12], is a two-step approach which first converts $H_2(z)$ into an intermediate all-pass system realizing an ordinary all-pole Lattice structure (the k 's), and next derive a set of weight factors (the α 's) used to scale a set of independent variables in this structure (the tapped signals) which, when added yield the numerator $B_2(z)$, i.e., the zeros of $H_2(z)$.

From Fig. 1, the computational complexity difference between the two structures is clearly seen to be in favor of DF-II. Further, it is apparent that for product summation conducted in fixed-point single-precision, the number of quantization errors (rounding or truncation) is higher for ITLS (the red dots). Similarly, for product summation calculated using double-precision, the DF-II also has fewer quantization errors (the green dots). Therefore, no matter the arithmetic approach used for product summation, the ITLS potentially introduces more quantization noise, thus being expected to have a lower output SNR as compared to DF-II. Consequently, from both the complexity- and the SNR-perspective, the DF-II might be considered the obvious choice for fixed-point implementation. We will demonstrate, however, that in the presence of AxC arithmetic, this is not necessarily an absolute conclusion.

III. APPROXIMATE MULTIPLICATION

We are interested in a 2's complement multiplier with variable word length d_m , and we have opted for a multiplication scheme based on the Radix-4 Booth algorithm, [13]. For a d_m -bit multiplicand X and a d_m -bit multiplier Y , a $2d_m$ -bit product $P = X \cdot Y$ is calculated. All three numbers, X , Y , and P are expressed in 2's complement number representation. We assign the MSB the index number 0, thus writing Y as

$$Y = -y_0 + \sum_{j=1}^{d_m-1} y_j \cdot 2^{-j} \quad (5)$$

For a dynamic range equal to $[-1; 1[$, the fixed point is located after the sign bit y_0 , and thus P can be expressed as

$$P = -y_0 \cdot X + \sum_{j=1}^{d_m-1} (y_j \cdot X) \cdot 2^{-j} \quad (6)$$

which next can be rewritten to

$$P = \frac{1}{2} \cdot \sum_{j=0}^{\lceil \frac{d_m}{2} \rceil - 1} X \cdot z_j \cdot 4^{-j} \quad (7)$$

$$z_j = y_j + y_{j+1} - 2 \cdot y_{j-1}; \quad z_j \in \{0, \pm 1, \pm 2\} \quad (8)$$

Equ. (7) and (8) are the fundamental operations of the multiplication where the $\lceil \frac{d_m}{2} \rceil$ Partial Products (PP) are sequentially shifted two bit positions against each other, i.e., Radix-4, and the final product is derived after a 1-bit right-shift of the sum.

Based on Equ. (7) and (8), the multiplier applied in this work is denoted the "Broken Booth Multiplier" (BBM), [14]. It represents an excellent compromise between execution time and power consumption, against *Mean Relative Error Distance*, a metric often used for evaluating the performance of AxC circuits, [4].

In [14], two variants of the BBM are discussed; Type_0 and Type_1. In our work, we employ Type_0, which is a multiplier that calculates all PPs exact before conducting sign-extension and addition (using 2's complement representation), Fig. 2. This scheme is applied no matter the sign of the individual PPs, which is different in the Type_1 approach where some of the negative PPs are alternatively represented as 1's complement numbers, thus saving the addition of an LSB.

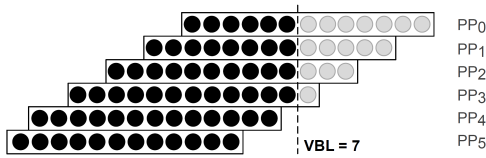


Fig. 2. The BBM Type_0 illustrated for word length 12x12 bit. The VBL-line defines a boundary where all bits to the right are nullified, thus eliminating addition of these bits, but leading to an approximate product, [14].

Note from Fig. 2 that the PPs, according to Equ. (7), are subsequently left-shifted two bit positions according to their individual numerical weighting. Furthermore, the PP word

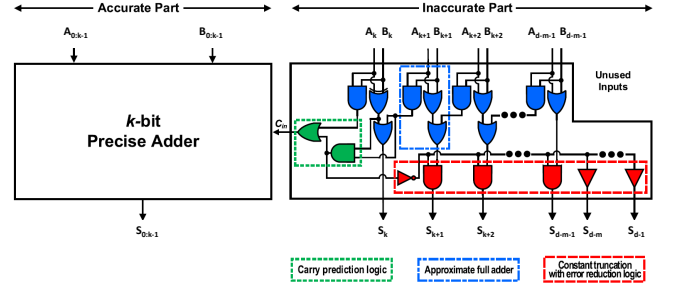


Fig. 3. A d_a -bit (for simplicity denoted as d) ERCPAA consists of an accurate k -bit MSB part, and an inaccurate $d_a - k$ bit LSB part. The latter is divided into an m bit unused section at its LSB-end (all bits are constant "0" or "1"), and a $d_a - k - m$ bit approximate MSB section. A speculative carry is fed from the inaccurate into the accurate part. Bit indexing from 0 to $d_a - 1$ which is opposite to the original work having index $n - 1$ to 0. Based on [15].

length equals $d_m + 1$ bit which is a consequence of the potential multiplication with ± 2 , according to Equ. (8). Also illustrated in Fig. 2 is a dotted vertical line denoted the *Vertical Breaking Level* (VBL). This line indicates the boundary between the exact and the approximate part of the multiplier, the exact part being to the left of the line. The bits situated on the right side, however, are all forced identically equal to zero which rules out the necessity for conducting addition. Due to the nature of the 2's complement number representation, for $VBL > 0$ the BBM Type_0 generates an Error Distance which is Gaussian distributed and negatively biased, the mean and standard deviation being functions of d_m and VBL, [9].

The experiments conducted in this work rely on a simulation model of a $d_m \times d_m$ bit Type_0 BBM. In addition to d_m being an adjustable parameter, the VBL value can be specified in the interval $[0; 2d_m - 1]$. The two operands X and Y are both considered being numerically less than 1, and are fed into the multiplier as floating point numbers, initially being converted into d_m -bit 2's complement representations. As indicated in Equ. (7), a total of $\lceil \frac{d_m}{2} \rceil$ 2's complement PPs are next derived. Starting with the least significant PP, the bits are then nullified from the LSB and up to the given VBL value. These modified PPs are now converted back into floating point numbers and numerically adjusted according to their weight factors. Finally, these weighted PPs are added, and the sum is right-shifted one bit position, providing the product P as a floating point number in the interval $[-1; 1[$, and with a $2d_m$ -bit 2's complement accuracy.

IV. APPROXIMATE ADDITION

We use a recently published AxC adder denoted the Error Reduced Carry Prediction Approximate Adder (ERC-PAA), [15], shown in Fig. 3. According to the authors, it has better or comparable performance when measured against a variety of other AxC adders considering the following three metrics: *i*) carry prediction error rate, *ii*) mean relative error distance, and *iii*) normalized mean error distance as compared to power, energy, and area-delay product.

In brief, the ERCPAA works as follows. An inaccurate part contains three functions denoted *i*) Approximate Full Adder

(blue), *ii*) Carry Prediction Logic (green), and *iii*) Constant Truncation With Error Reduction Logic (red). The Approximate Full Adder elements replace traditional accurate Full Adders (FA) which conduct the following 3-input operations

$$sum = (a \oplus b) \oplus c_{in} \quad (9)$$

$$carry = (a \cdot b) + c_{in} \cdot (a \oplus b) \quad (10)$$

In the ERPCAA, the $a \oplus b$ operation in the sum calculation, Equ. (9), is simplified to a Boolean $a + b$ function, the exception being the most significant bit which is implemented as an exclusive OR of the two input operand bit to maintain the accuracy at the MSB-end of the inaccurate part.

The $carry$ calculation, i.e., carry generate and propagate, Equ. (10), has been reduced to a single generate, i.e., a Boolean $a \cdot b$ operation. This eliminates the longitudinal carry chain of a traditional Ripple Carry Adder (RCA). However, the ERPCAA supports a simple, yet effective carry scheme which spans two consecutive bits, i.e., the carry bit generated at position j is logically added to the sum bit generated at position $j - 1$. A carry generated in the MSB-end of the inaccurate part is fed into the accurate part as an ordinary $carry_{in}$ signal. This carry is based on *i*) the carries from the two most significant bits, and *ii*) the MSB sum. Using these three bits (whereas using only a carry generated from the MSB), reduces the error in the carry prediction at the input to the accurate part, but still provides a reduced gate count as compared to a full-size RCA.

The ERPCAA also introduces a mechanism to control all the sum bit from index $k + 1$ down to $d_a - 1$. The control signal, calculated as $\overline{sum_k} \cdot \overline{carry_{k+1}}$ from the Approximate FAs, is equal to "1" when one or both of these bit equals "0". In the interval from $(k + 1)$ down to $(d_a - m - 1)$, the control signal, when equal to "1", is used to pass the OR-combined sum- and carry-bit from the Approximate FAs directly to the sum output bit, s_i . Similarly, if the control signal equals "0" it forces all sum bits in this specific interval equal to "0" to compensate numerically for the s_k bit which in this particular situation equals a faulty "1". From bit position $d_a - m$ down to $d_a - 1$, the control signal is fed directly to the sum output bit, s_i , thus generating an m -bit constant output. This eliminates the need for Approximate FAs in the LSB-end of the adder's inaccurate part.

In our simulation model, d_a , and the AxC parameters k and m can be tuned individually. The augend A and the addend B , as well as the resulting sum S , are all floating-point numbers which are converted to/from 2's complement representation.

V. EXPERIMENTAL RESULTS AND DISCUSSION

The evaluation of the two filter structures is based on a series of four experiments which expose their time domain behaviour in terms of impulse responses (i.e., all frequencies are evaluated). The filters are excited with the signal $s \cdot \delta[n]$, where $n \in [0; N - 1]$, and s is a scalar which introduces a safe compromise between utilization of the 2's complement dynamic range $[-1; 1]$ and the elimination of numerical overflow in the variables where overflow is not allowed (i.e., at all

green variables in Fig. 1). We measure the AxC-based impulse response from each structure, $h_{AxC,DF}[n]$ and $h_{AxC,ITLS}[n]$, using a d_m -bit BBM and a d_a -bit ERCPAA, and next subtract each of these sequences from the exact response $h_{exact}[n]$ derived from a 64-bit floating-point implementation of the filter. These resulting sequences are then squared and summed to obtain the residual variance

$$\sigma_x^2 = 10 \cdot \log \sum_{n=0}^{N-1} (h_{exact}[n] - h_{AxC,x}[n])^2 \quad (11)$$

where $x = [DF, ITLS]$, and where $N = 1000$ is empirically found adequate for the impulse responses to reach their steady state. This metric indicates (in dB) the deviation between the exact filter and the inexact counterparts for varying r , θ_ℓ , and the BBM and ERCPAA AxC-parameters.

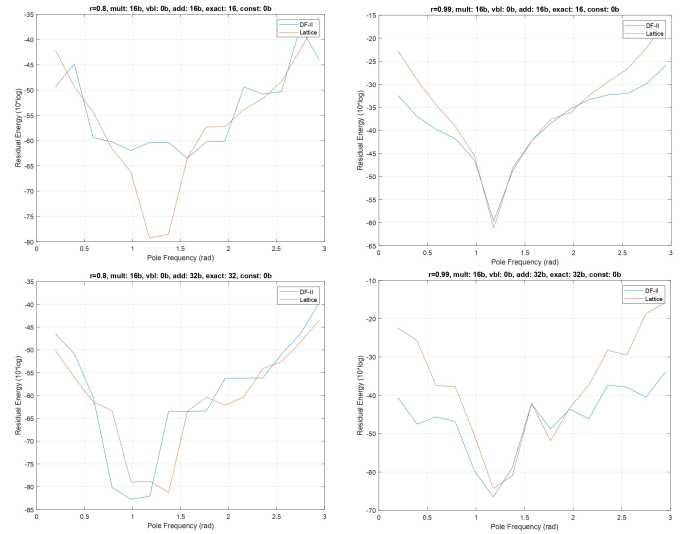


Fig. 4. The residual variance in dB for circuits without approximation. In this, and all following plots, the upper row represents single-precision addition, i.e., $d_m = 16$ and $d_a = 16$, while the lower row is double-precision, $d_m = 16$ and $d_a = 32$. The two columns represent $r = 0.8$ and $r = 0.99$, respectively. The abscissa is the pole-frequency θ_ℓ in radians. The blue curve shows DF-II, and the red one illustrates ITLS.

The huge number of possible combinations caused by the many adjustable parameters makes it impossible to explore the complete solution space. Therefore, for both structures we investigate four major scenarios; $r = 0.8$ and $r = 0.99$ combined with 16-bit multiplication followed by *i*) single- and *ii*) double-precision addition, i.e., 16- and 32-bit addition, respectively.

Our first experiment uses no approximation, i.e., $VBL = 0$, and $k = d_a$, Fig. 4. Note the expected form of all graphs; when the pole-frequency approaches the zero-frequency, the pole-dependent amplification is (partly) eliminated, increasing the numerical tolerance of both structures, leading to the dip in the vicinity of the zero (i.e., $\omega \approx \omega_{zero}$). For $r = 0.99$ the shape of the dip is more marked which is also expected due to the more sudden cancellation of the pole-effect when ω sweeps by ω_{zero} . It is noted that for $r = 0.99$, σ_x^2 is generally larger than for $r = 0.8$. This is also expected since

the filter's Q -factor increases for $r \rightarrow 1$, thus reducing the numerical robustness, [10]. For frequencies in a distance of more than $\pi/3$ rad from ω_{zero} , we observe for $r = 0.99$ that $\sigma_{ITLS}^2 > \sigma_{DF}^2$, both for single- and double-precision. This is also expected and is explained by the higher number of quantization errors in the ITLS as compared to the DF-II. This effect is not seen for $r = 0.8$ where $\sigma_{ITLS}^2 \approx \sigma_{DF}^2$ for all frequencies (except for $\omega \approx \omega_{zero}$, $r = 0.8$, single-precision, where ITLS is significantly better), which we explain by the increased numerical tolerance for the low- Q filters.

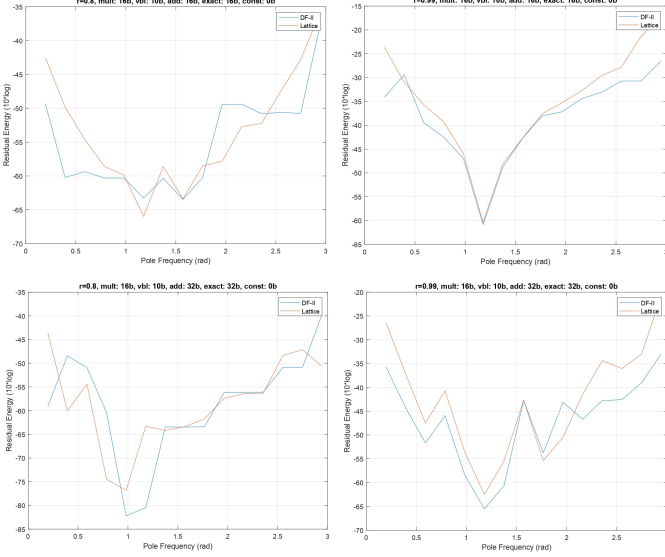


Fig. 5. σ_x^2 illustrated for exact addition and multiplication with VBL = 10.

In our second experiment, we use an AxC multiplier and an exact adder. We opt for VBL = 10, i.e., the 10 least significant bits of the 32-bit products are nullified prior to product summation, Fig. 5. The overall shape of the performance curves are generally maintained, but there are at least two important observations. First and foremost we note that for $r = 0.99$ in both single- and double-precision, the better performance of DF-II for the “no AxC” case has been reduced, in particular for single-precision. In large this is due to a deteriorated DF-II performance, which indicates that for high- Q filters, the ITLS is more resistant towards the inaccuracy introduced by the AxC multiplier. We explain this by the $\pm k_i$ multiplications in the ITLS, which generate products with numerical offsets of different sign, potentially being out-balanced in the following product summation. Secondly, for $r = 0.8$, single-precision, at $\omega \approx \omega_{zero}$ both structures show $\sigma_x^2 \approx -60$ dB. It is the ITLS performance which has degraded as compared to the “no AxC” scenario, thus indicating that closely spaced poles and zeros eliminates somewhat the above mentioned $\pm k_i$ multiplication effect of the ITLS. We consider this being caused by the reduced numerical values of the internal variables in the non-recursive part of the filter for $\omega \approx \omega_{zero}$. This observation is similarly strengthened in the results for $r = 0.8$, double-precision, and $\omega \approx \omega_{zero}$ where ITLS, as compared to the “no AxC” case, has its performance degraded from ≈ -80 dB

to ≈ -75 dB, while DF-II maintains a better than -80 dB performance for identical parameters.

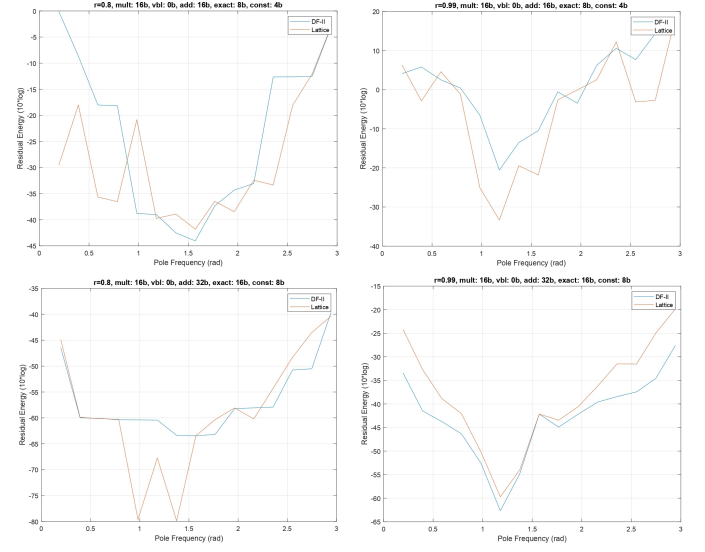


Fig. 6. σ_x^2 for exact multiplier, and adder with $k = d_{add}/2$, $m = d_{add}/4$.

Our third experiment involves an exact multiplier, i.e., VBL = 0, and an AxC adder. We opt for a word-length of the accurate part equal to half the total word-length of the sum, i.e., $k = d_{add}/2$, and we choose half of the inaccurate part to have a constant value, i.e., $m = d_{add}/4$, Fig. 6. Some noteworthy findings emerge from this setup. Most remarkable is the fact that the ITLS, except for the high- Q double-precision scenario, is now mostly better than or comparable to the DF-II, despite the 50% higher adder count, and thus AxC-related noise, in the ITLS. Apart from some few variations in the superior ITLS-performance, the overall indication is that the ITLS is able to withstand better the substantial inaccuracy being introduced by the AxC additions. Again, we expect this being partly due to the $\pm k_i$ multiplications in the feedback loop of the ITLS, which recursively balances out (some of) the introduced ERPCAA errors. It should be noted, however, that there is a marked performance difference between the single- and the double-precision scenarios. For double-precision and $r = 0.8$ we found a significantly better performance for both structures, seen as -45 dB for single-precision versus -80 dB for double-precision in best case. This observation indicates that reducing the accuracy of the least significant half of the sum when performing AxC addition is relatively harmless for double-precision, whereas more precaution is needed when working with single-precision. For high- Q filters, the situation is even more pronounced. For DF-II, the result is -20 dB versus -62 dB at best for single- and double-precision, respectively. For ITLS, the corresponding numbers are -32 dB and -60 dB, clearly indicating that the ITLS is the preferred structure for single-precision high- Q filters.

Our fourth and final experiment involves simultaneous application of approximation in the multiplier, i.e., VBL = 10, and in the adder, i.e., $k = d_{add}/2$ and $m = d_{add}/4$,

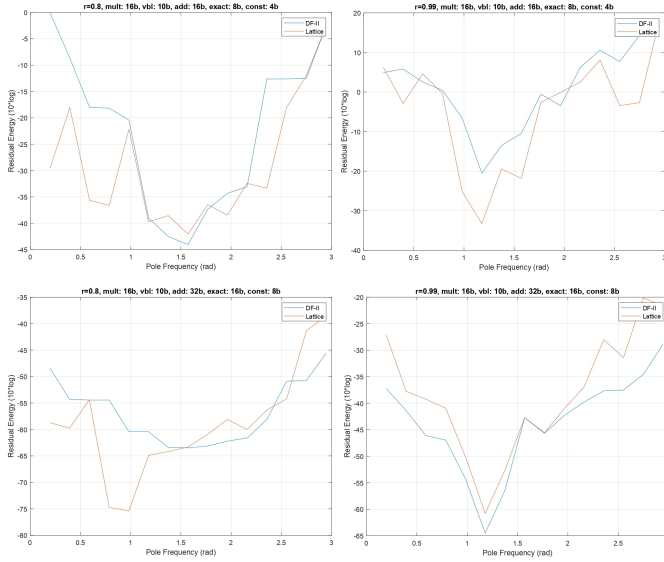


Fig. 7. σ_x^2 for full AxC; VBL = 10, and $k = d_{add}/2$, $m = d_{add}/4$.

Fig. 7. The immediate observation is the relatively limited deviation/degradation of performance for all four scenarios as compared to the setup with only AxC addition, Fig. 6. Except from some few cases, the eight curves essentially have the same progress. To a large extent, they also have the same performance measure, the exception being *i*) DF-II for $r = 8$ and single-precision, *ii*) DF-II for $r = 8$ and double-precision, and *iii*) ITLS for $r = 8$ and double-precision, which all show a slight deterioration for low frequencies as compared to the third experiment. This consistent pattern very clearly indicates that in a combined AxC multiplier/adder configuration, the AxC adder is by far the most critical component (for the parameter settings chosen in this study). Additionally, for this combined experiment, we conclude that choosing the ITLS over the DF-II seems to be an advantageous solution in terms of improved numerical performance in an AxC scenario, the exception being for high- Q filters implemented in double-precision. For this particular combination, it appears that the DF-II almost always performs better.

VI. CONCLUSION

We have investigated AxC multiplication and AxC addition applied to two types of implementation structures for recursive 2^{nd} order digital filters, the DF-II and the ITLS. Based on the Radix-4 BBM Type_0 multiplier and the ERPCAA adder, both being reported among the circuits providing the best trade-off between accuracy and resource reduction, we have conducted an experimental campaign involving adjustable parameters associated to the circuits and the filters. In essence, our findings are that *i*) approximate multiplication and addition can be used (individually or combined) in the two selected structures without compromising the stability of the filters, *ii*) for AxC multiplication, exact addition, and high- Q filters, the ITLS is more resistant towards reduced product accuracy, despite closely spaced poles and zeros tend to eliminate this effect, *iii*)

for exact multiplier and AxC adder, the ITLS is mostly better than or comparable to the DF-II, meaning that the ITLS can withstand better the inaccuracy introduced by AxC additions, the ITLS being the preferred structure for single-precision high- Q filters, *iv*) some exceptions (slight deterioration for low frequencies) highlight that in a combined AxC multiplier/AxC adder configuration, addition is the most critical component, and *v*) in such a combination, the ITLS seems to be more advantageous in terms of numerical performance as compared to the DF-II, except for high- Q filters implemented in double-precision, for which the DF-II almost always performs better.

The number of parameters and their many possible settings lead to a solution space for the investigated problem which is sized way beyond what is possible to explore manually. Many more and insightful experiments could therefore be conducted by developing an automated solution space exploration framework, potentially unveiling sets of parameter combinations which represent local optimal solutions. Besides, since we have clearly demonstrated the viable possibility for such a match, our future investigations will focus on the amount of resource reduction which can be achieved as compared to using traditional multiplier and adder circuits.

REFERENCES

- [1] Y. Sun, G. Wang, B. Yin, J. R. Cavallaro, and T. Ly, *High-level Design Tools for Complex DSP Applications, Chapter 8 in "DSP for Embedded and Real-Time Systems"*. Elsevier Inc., 2012.
- [2] M. Bertamini and M. Kubovy, *Human Perception*. Routledge, ISBN 9781138355972, 2022.
- [3] S. Mittal, "A survey of techniques for approximate computing," *ACM Computing Surveys*, vol. 48, no. 4, pp. 62:1–62:23, 2016.
- [4] H. Jiang, F. J. H. Santiago, H. Mo, L. Liu, and J. Han, "Approximate arithmetic circuits: A survey, characterization, and recent applications," *Proceedings of the IEEE*, vol. 108, no. 12, pp. 2108–2135, 2020.
- [5] M. Pashaeifar and M. Kamal, "A theoretical framework for quality estimation and optimization of dsp applications using low-power approximate adders," *IEEE Trans. on Circuits and Systems-I: Regular Papers*, vol. 66, no. 1, 2019.
- [6] —, "Approximate adder synthesis for area- and energy-efficient fir filters in cmos vlsi," *IEEE 13th Int. New Circuits and Systems Conference*, 2015.
- [7] W. Hui, G. Chang, V. Gormathi, R. Valarmathi, V. S. Balaji, and V. Elamara, "Revisiting fpga implementation of digital filters and exploring approximate computing on biomedical signals," *Jour. of Medical Imaging and Health Informatics*, vol. 10, no. 9, pp. 2020–2004, 2020.
- [8] R. Pilipovic, V. Risojevic, and P. Bulic, "On the design of an energy efficient digital iir a-weighting using approximate multiplication," *Sensors*, vol. 21, no. 732, 2021.
- [9] P. Koch, J. Østergaard, and O. Andersen, "On numerical robustness of bi-quad structures using fixed-point approximate multiplication," *Proc. of the 25th Int. Symposium on Wireless Personal Multimedia Communication*, pp. 226–231, 2022.
- [10] S. K. Mitra, *Digital Signal Processing, A Computer-Based Approach*. McGraw-Hill Int. Edition, ISBN 0-07-118175-X, 2001.
- [11] D. Schlichthärle, *Digital Filters, Basics and Design, 2nd Ed.* Springer, ISBN 978-3-642-14324-3, 2011.
- [12] J. D. Markel and J. A. H. Gray, *Linear Prediction of Speech*. Springer-Verlag, Berlin Heidelberg New York, ISBN-13: 978-3-642-66288-1, 1976.
- [13] B. Parhami, *Computer Arithmetic, Algorithms and Hardware Designs*. Oxford University Press, 2000.
- [14] F. Farshchi, M. S. Abrishami, and S. M. Fakhrarie, "New approximate multiplier for low power digital signal processing," *Proc. 17th Int. Symp. on Computer Architecture and Digital Systems*, pp. 25–30, 2013.
- [15] J. Lee, H. Seo, H. Seok, and Y. Kim, "A novel approximate adder design using error reduced carry prediction and constant truncation," *IEEE Access*, vol. 9, pp. 939–953, Sep. 2021.