Compact Fixed-Point Filter Implementation

Timur I. Karimov, Denis N. Butusov, Valerii S. Andreev, Vyacheslav G. Rybin, Dmitry I. Kaplun
St. Petersburg Electrotechnical University "LETI"
St. Petersburg, Russia
tikarimov, dnbutusov@etu.ru

***Abstract*—The main difficulty in IIR-filter hardware implementation using fixed-point arithmetic is an accurate continuous to discrete model conversion. We propose an alternative approach to IIR-filters fixed-point implementation based on adaptive discrete operator selection (z-operator or δ-operator) and filter parameters optimization. This approach provides significant reduction of utilized logic elements for the given level of implementation accuracy.**

I. Introduction

Fixed-point arithmetic is widely used in custom hardware designs, digital signal processors and microcontroller programs implementation. The advantages of operations with short numbers (up to 32 bits) are especially evident when target platform is FPGA (field programmable gate array) or ASIC (custom chip). The literature provides various data on relative effectiveness of fixed-point solutions, but it always notes the lower expense of the chip logic cells, the increase of maximum clock frequency of the device and the reduction of energy consumption. Govindu et al. [1] state that fixed-point arithmetic allows to reduce the energy consumption of device up to 7–15 times, 5–10 times reducing the required chip area and 1.25 times increasing the performance. Ewe et al. [2] show that infinite impulse response (IIR) filter implementation in 32-bit fixed-point arithmetic is 5.7 times more profitable considering LEs utilization. The filter response time can be decreased 5.2 times compared to the IEEE 754 (single) data type. Over the past decade an efficiency of the hardware solutions for floating-point operations has grown rapidly. However, many up-to-date publications [3], [4] still note multiple superiority of the fixed-point arithmetic for hardware implementation.

In general, increasing a filter accuracy and chip space economy are opposite problems. Therefore it is necessary to find their optimal ratio for the filter design task. In this paper we use the term «compactness» to specify the relation between logic elements number and root-mean-square or maximum error of hardware filter response. Considering examples in [2], the most precise fixed-point filter can be three times more compact than a filter operating with numbers in IEEE 754 format. Therefore, when the compactness of filter implementation becomes a design efficiency criterion the fixed-point arithmetic is preferable.

Ways to increase filters compactness can be divided into three main groups: use of alternative number representation formats, special mathematical methods application and model optimization. The first group includes use of variable dynamic range number format (for example, dual fixed-point) [2], stochastic computing, modular arithmetic etc. The second group includes application of higher order continuous-to-discrete time conversion methods [5], optimal implementation forms [6] and alternative discrete operators [7]–[9]. An optimization approach includes tuning of filter amplification factor to avoid overflow and the choice of word size for the state variables and filter coefficients representation. Currently, approaches from the third group are common, while the methods of the first two groups are used relatively rarely despite their efficiency in many cases. From one hand, constant growth of FPGA logic cells number mitigates the severity of the filter compaction problem. But in many cases the most compact implementation is still the most preferable, e.g., in the aerospace industry tasks. In our work we show that of alternative discrete operators application in combination with optimization techniques is the efficient way to obtain compact filter implementation on FPGA.

The paper is organized as follows. Section II specifies discrete δ-operator and its modifications, as well as operator choice criterion. Section III provides filter coefficients tuning technique based on numerical-analytical approach. Section IV describes several practical examples. The paper ends with conclusion.

II. Alternative discrete operators

One of the alternative approaches to filter design consists in the replacement of the discrete z-operator with so-called δ-operator [7]:

$$\delta \equiv \frac{z-1}{T_s} \tag{1}$$

where T_s is a sampling period. When sampling period decreases, zeros and poles of the discrete δ-filter converge to zeros and poles of the continuous prototype. δ-operator expressed with equation (1) had not have any parameters, thus parametric δ-operator was suggested later [8]:

$$\delta \equiv \frac{z-1}{\Delta}. \tag{2}$$

Here Δ is a free parameter. To decrease a quantization noise and to simplify hardware implementation of operator δ^{-1} it was proposed to define parameter Δ as negative degree of number 2.

ISSN 2305-7254

TABLE I. EQUATIONS FOR Z →Δ COEFFICIENTS RECALCULATION

β_0	$\beta_0 = b_0$	α_0	$\alpha_0 = 1$
β_1	$\beta_1 = \frac{2b_0 + b_1}{\Delta}$	α_1	$\alpha_1 = \frac{2 + a_1}{\Delta}$
β_2	$\beta_2 = \frac{b_0 + b_1 + b_2}{\Delta^2}$	α_2	$\alpha_2 = \frac{1 + a_1 + a_2}{\Delta^2}$

One can see that only two additional summations are added to the structure of the 2nd order section of δ-filter compared to the structure of z-filter, see Fig. 1. Multiplication in FPGA is implemented as a simple commutation of a binary bus with discarding low-order bits.

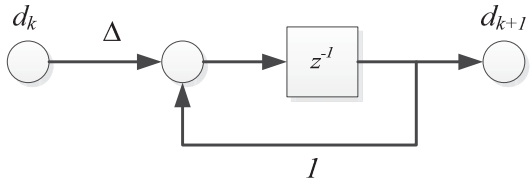


Fig. 1.Implementation of δ^{-1} operator

To implement a filter using δ-operator it is necessary to convert the coefficients of z-model using the equations from Table I. These expressions can be easily obtained via substitution of (2) in the discrete section of the 2nd order:

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}. \quad (3)$$

Here z denotes shift operator. The second order section based on δ-operator looks as follows:

$$H(\delta) = \frac{\beta_0 + \beta_1 \delta^{-1} + \beta_2 \delta^{-2}}{1 + \alpha_1 \delta^{-1} + \alpha_2 \delta^{-2}}. \quad (4)$$

The Direct Form I is not suitable for the implementation of δ-filter because of the unstable pole $z^{-1} = 1$ in the structure of operator δ^{-1} (Fig. 1) [8]:

$$\delta^{-1} = \frac{\Delta z^{-1}}{1 - z^{-1}}.$$

Therefore, δ-systems are usually implemented in the Direct Form II (DFII) or its transposed version (DFIIt), see Fig. 2.

Alternative discrete operators allow resolving two main challenges connected with the limited precision of number representation using fixed-point arithmetic. It includes the following:

1) Avoiding the filter zeros and poles "sticking" together and with the z-plane unit and thus making the filter coefficients more informative;

2) Filter state variables amplitudes equalization. For this reason the choice $\Delta = T_s$ is not optimal in the most cases. Usually $\Delta >> T_s$ gives better results.

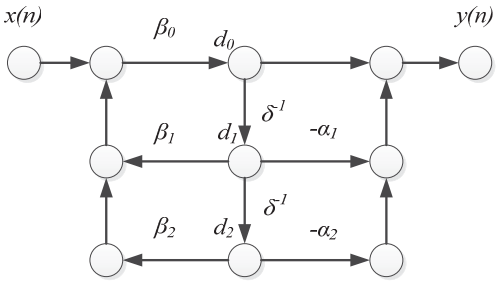


Fig. 2. Representation of the Direct Form II for the 2nd order section. d_i denote the state variables

Fig. 3 represents the frequency responses for various state variables of two band-stop filter implementations

$$H(s) = \frac{s^2 + \omega_0^2}{s^2 + \frac{\omega_0}{Q} s + \omega_0^2}, Q = 1, \omega_0 = 2\pi \cdot 440 Hz \quad (5)$$

Both filters have the sampling rate equal to 44.1 kHz.

It is generally assumed that amplitudes alignment for the state variables means alignment of their H_∞-norms. When implementing a filter in DFII, the equation for choosing the quasi-optimal value of Δ^* for every 2nd order section is [8]:

$$\Delta^* = \max\left(\frac{\left\| \Delta^{-1} F_{\delta,1}(z) \right\|_\infty}{\left\| \Delta^{-2} F_{\delta,2}(z) \right\|_\infty}; \sqrt{\frac{\left\| F_{\delta,0}(z) \right\|_\infty}{\left\| \Delta^{-2} F_{\delta,2}(z) \right\|_\infty}} \right) \quad (6)$$

where $F_{\delta,i}(z) = d_{\delta,i} / x$ is the transfer functions from input to the state variables:

$$F_{\delta,0}(z) = \frac{(1 - z^{-1})^2}{1 + a_1 z^{-1} + a_2 z^{-2}},$$

$$F_{\delta,1}(z) = \frac{\Delta z^{-1}(1 - z^{-1})}{1 + a_1 z^{-1} + a_2 z^{-2}},$$

$$F_{\delta,2}(z) = \frac{\Delta^2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}.$$

Expressions for DFIIt look similar and give close values of Δ^* in practice. Having Δ^* it is necessary to choose

$$\Delta = 2^{-n} \approx \Delta^*, n \in N.$$

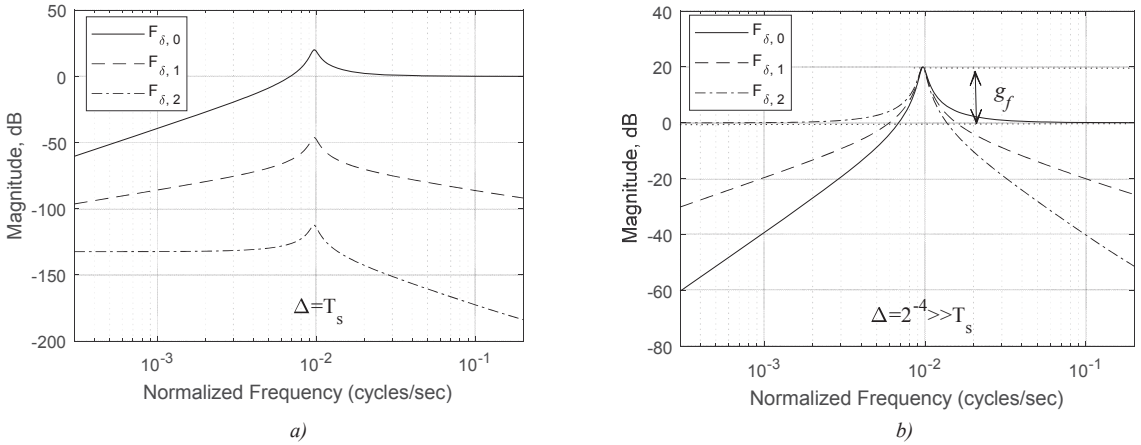


Fig. 3. Frequency responses for the state variables of band-rejection filter (5) for various values of Δ: a) $\Delta = T_s$ b) $\Delta >> T_s$

It is impractical to reject z-operator completely because δ-operator complicates the filter structure and some negative effects can appear in δ-model at low sampling frequencies, e.g. limit cycles. Besides, δ-model accuracy is not superior to accuracy of z-model on these frequencies.

The criterion for an operator choice was introduced by the authors of this paper earlier in [10]. Considering this criterion, the implementation of the 2nd order section with complex conjugate poles $r_{1,2} = \sigma \pm j\omega$ will be more accurate using δ-operator when

$$(\sigma T_s)^2 + (\omega T_s)^2 < 1. \quad (7)$$

In other case one should prefer z-operator.

When a cutoff frequency and the filter type are known in advance a simple criterion can be used:

$$\lambda < 0.1 \quad (8)$$

where $\lambda = f_{cutoff} / f_s$. Criterion (8) is applicable for the low-pass filters, band-stop filters and for several other cases when it is in a good agreement with criterion (7).

III. FPGA IMPLEMENTATION OF THE FIXED-POINT FILTERS

The key feature of the fixed-point algorithms development for FPGA implementation is the flexibility of word lengths selection. Thus the task of filter design can be reduced to choosing the minimum bit depth values which provide the required accuracy level.

We briefly determine our filter implementation strategy below. The filter of the arbitrary high order specified in arithmetic with a floating point should be divided on the 1st and 2nd order sections. Each section should be analyzed considering discrete operator choosing criteria (7) or (8). Then sections are sampled with a given period T_s using bilinear transform. Further we use an iterative algorithm of filter conversion to the fixed-point data type. The algorithm of δ-filter implementation is as follows.

1) First, we substitute $\Delta = 1$ in (6) and compute Δ^* in floating-point arithmetic. Then calculate

$$N_\Delta = \left\lceil \log_2 \Delta^* \right\rceil, n \in N$$

and find $g_f = \max\left(\left\| F_{\delta,i}(z) \right\|_\infty \right), i = \overline{0..2}$ (see Fig. 3), and then calculate

$$N_f = \left\lfloor \log_2 g_f \right\rfloor, n \in N.$$

2) Next, we set the initial word length for the state variables representation

$$WL = K + N_f + 1$$

where K is a bit depth of input signal (ADC/bus of the previous system section/part), and 1 is the overflow preventing bit.

Then we convert filter to the fixed-point arithmetic. The numerator and denominator coefficients should be scaled individually. The initial approximation of the fractional part length for the both cases is equal to $WL/2$.

3) We set the test linear chirp (LC) signal covering the passband and stopband of the filter. The convenience of the LC signal consists in the predictability of its spectral characteristics as well as in the opportunity to construct frequency response using envelop response of an integer filter faster and more precisely than by Fourier transform. Then we should calculate a reference filter response with real coefficients U_{ref}.

4) We simulate an integer filter considering the rounding of the coefficients and other effects related to the integer arithmetic: shift to the fractional part of the filter coefficients after multiplication, overflows such as “wrap around” etc.

If there are no overflows and difference between the output signal and reference doesn’t exceed the required level

$$\forall i : \max\left|U_{out}^{i} - U_{ref}^{i}\right| < \varepsilon$$

then the required implementation is found.

At each step the following parameters are randomly varied using Monte Carlo method:

- $n = N_{\Delta} \pm 1$ $\Delta = 2^{-n}$;
- fractional part length of the numerator and denominator;

To increase the accuracy of an input signal representation and to fulfill whole width of the state variables we implement left circular shift of the ADC signal (Fig. 4) in the range

$$M \in \overline{0...WL - K - N_f - 1.}$$

Than we calculate the response of the integer data type filter for each parameters variation and increment the machine word length if the parameters combination providing required accuracy level without overflows was not found after N_{iter} iterations.

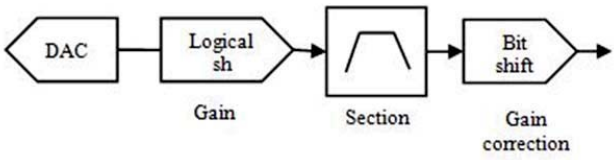


Fig. 4. Scaling scheme for an input signal to increase the accuracy of signal internal representation

5) On step 5 we test the found integer filter on the signal set including white noise with various amplitude. During this the widths of internal buses between state variables, adders and multipliers are finally determined. One should go back to the step 4 if the overflow of the state variables is detected.

6) Finally, we substitute numerical parameters of the filter into a code template of the filter module and its testbench in Verilog HDL. Also we generate a set of files for the input test signals. The synthesized filter module can be included in the design of target device without re-verification because of the debugged automatic code generation procedure.

In case of z-operator implementation some specific steps should be excluded, but general idea is the same.

IV. Experimental findings

We carried out two series of numerical experiments. In the first experimental set we obtained a quantitative estimation of FPGA hardware resource consumption while implementing z-filters and δ-filters. One of the theoretically predicted results was the confirmation of the correctness for criteria (7) and (8).

The second set of experiments was devoted to the research of some aspects of synthesis algorithm. We considered the optimal filter representation form selection and the required iterations number N_{iter} estimation of Monte Carlo process.

A. Comparison of discrete operators

We investigated several implementations of filter (5) with various sampling rates. When implementing 9-bit filters, it was assumed that ADC bit depth is 8 bit, for 12-bit filters ADC depth was set to 10 bit, and for 16-bit filters we set bit depth equal to 12 bit. These values of the bit depth are typical for common industrial ADC and are widespread through many real devices and applications. Experimental results are provided in Fig. 5(*a*). The chart shows which implementation is the most accurate, assuming those can be constructed using the above strategy.

Errors in filter implementations using z- and δ-operators are almost equal at point $\lambda = 0.1$. It is in good agreement with our theoretical assumptions. Note that while λ decreases, the error of both discrete implementations for a fixed machine word length increases.

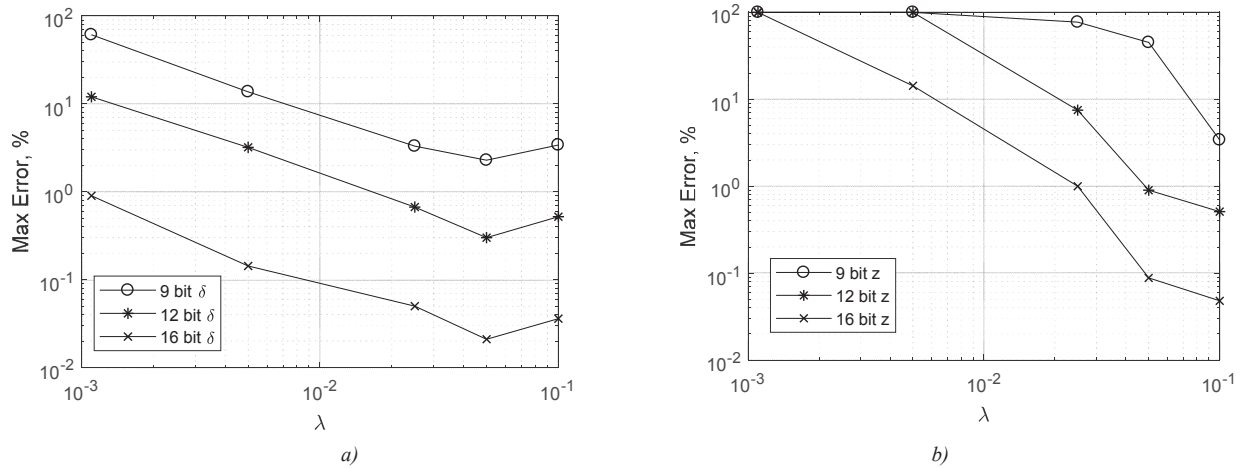


Fig. 5. Errors of the discrete filters for various operator types depending to the normalized cutoff frequency and word lengths

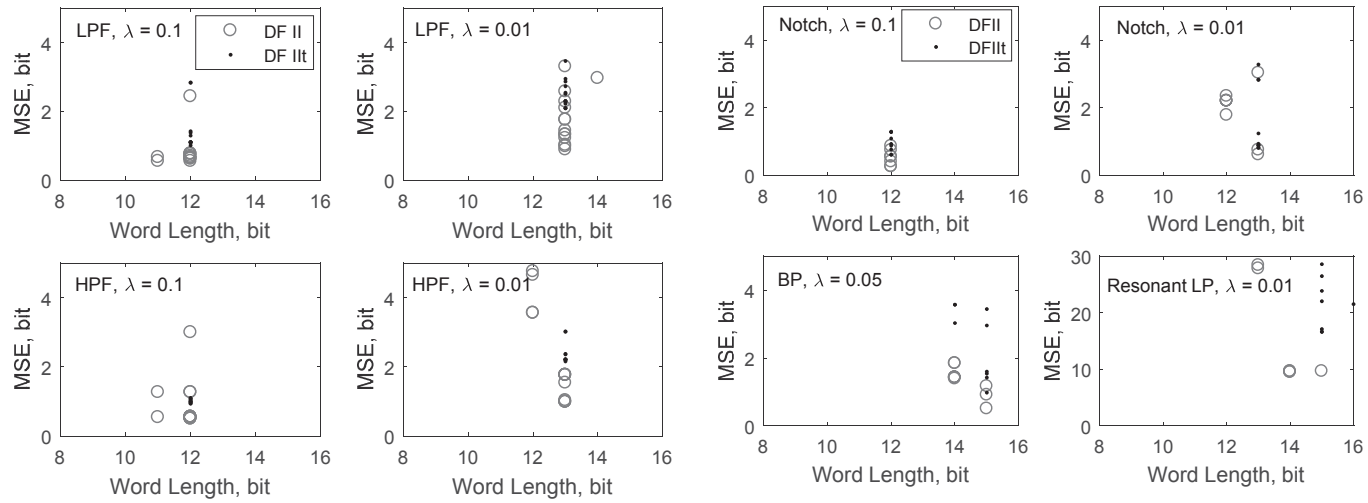


Fig. 6. Comparison of δ-implementations of various filters in DFII and DFIIt forms. Here LPF denotes low-pass filter, HPF – high-pass filter, BP – band-pass filter, Notch – band-stop filter, and Resonant LP is an LPF with 20 dB peak resonance

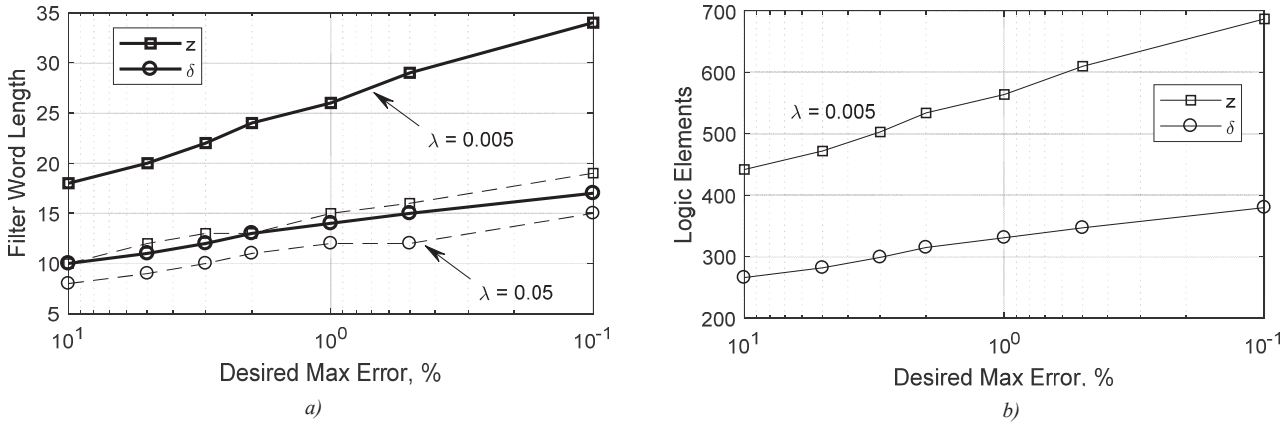
Fig. 5(*b*) shows the approximation of the experimental results. The represented charts clearly demonstrate the advantages of δ-operator when $\lambda \to 0$.

DFII structures were implemented in Verilog HDL and then synthesized with Quartus II software. We used Altera Cyclone IV target without embedded multipliers utilization. Quantitative estimations of logic elements (LE) consumption are shown in Fig. 7. One can see that if we want the maximum error do not to exceed level $\varepsilon = 0.1\%$, z-operator filter requires twice as many logical elements. If λ or ε decreases the benefit of using δ-operator is even greater. The dependency between logic elements utilization and the word length is almost linear.

B. Investigation of the synthesis algorithm

To compare two representation forms of δ-filters under investigation – DFII and DFIIt – several different filters were implemented with various parameters (see Fig. 6). In these experiments we used standard deviation (SD) between fixed point filter and its floating point prototype response as a metric.

Since the filter optimization was performed for maximum error level ε, this level was approximately the same for both implementations. However, in almost all implementations DFII form is more advantageous in the terms of SD level. This do not completely agree with results of the earlier work [8], where a preference to the DFIIt form was given.



a) *b)*

Fig. 7. Word length (a) and logic elements consumption (b) for case of band-stop filter implementation using various discrete operators

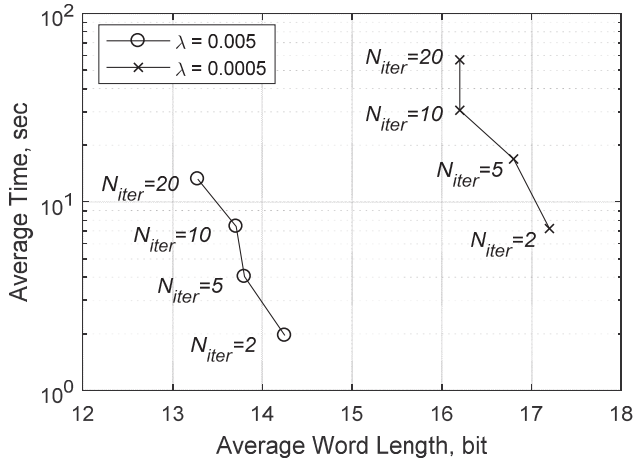


Fig. 8. The dependency between average search time (s) and found word length for different Monte Carlo iterations number

An example of search time vs Monte Carlo iterations number and word length analysis is given in Fig. 8. Increasing the number of iterations of the algorithm from 2-5 to 10-20 we can reduce word length for found implementation averagely by one bit. However, the algorithm execution time increases by an order. The duration of search is closely related to the necessity of low normalized frequencies signal simulation when $\lambda \to 0$.

V. Conclusion

We proposed an algorithm for fixed-point filter implementation based on adaptive discrete operator selection approach and Monte Carlo filter parameters tuning method, focusing in our paper on alternative δ-operator application technique. Our algorithm allows reducing the number of required logical elements significantly for FPGA implementation, up to 2-3 times and more depending on a task.

In our algorithm expected effect of discrete z or δ-operator application can be predicted using one of two simple criteria. For most tested δ-filters implementations the Direct Form II turned out to be more preferable, that disaccords with early works which established the transposed Direct Form II superiority.

In our further research we will examine an efficiency of alternative number representation formats for filter implementation. Also a study of the δ-operator application to FIR filters design will be accomplished.

Acknowledgment

The reported study was partially supported by RFBR, research project No. 17-07-00862.

References

[1] G. Govindu, L. Zhuo, S. Choi, P. Gundala, and V. K. Prasanna, “Area, and Power Performance Analysis of a Floating-point based Application on FPGAs”, *in Proc. HPEC Conf.,* Sept. 2003.

[2] C. T. Ewe, P. Y. K. Cheung, and G. A. Constantinides, “Dual fixed-point: an efficient alternative to floating-point computation”, *in Proc. Int. Conf. F. Program. Log.*, vol. 3203 of Le, Aug. 2004, pp. 200 – 208.

[3] K. Aruna and J. Bhaskararao, “Design and Implementation of Fixed Point and Floating Point PID Controllers in VIVADO HLS using FPGA”, *ISSN*, vol. 409, 2016.

[4] A. Finnerty and H. Ratigner, “Reduce Power and Cost by Converting from Floating Point to Fixed Point”, *WP491 (v1.0),* March 2017.

[5] A. M. Schneider, J. T. Kaneshge, and F. D. Groutage, “Higher order s-to-z mapping functions and their application in digitizing continuous-time filters”, *in Proc. IEEE*, vol. 79, no. 11, 1991, pp. 1661–1674.

[6] L. Jackson, A. Lindgren, and Young Kim, “Optimal synthesis of second-order state-space structures for digital filters”, *IEEE Trans. Circuits Syst.*, vol. 26, no. 3, Mar. 1979, pp. 149–153.

[7] R. H. Middleton and G. C. Goodwin, “Improved Finite Word Length Characteristics in Digital Control Using Delta Operators”, *IEEE Trans. Automat. Contr.*, vol. 31, no. November, Nov. 1986, pp. 1015–1021.

[8] J. Kauraniemi, T. I. Laakso, and I. Hartimo, “Delta Operator Realizations of Direct-Form IIR Filters”, *IEEE Trans. Circuits Syst. - II Analog Digit. Signal Process.*, vol. 45, no. 1, 1998, pp. 41–52.

[9] D. N. Butusov, T. I. Karimov, D. I. Kaplun, and A. I. Karimov, “Delta operator filter design for hydroacoustic tasks”, *in Proc. 6th Mediterranean Conference on Embedded Computing (MECO)*, June 2017, pp. 1–4.

[10] D. N. Butusov, T. I. Karimov, D. I. Kaplun, A. I. Karimov, Y. Huang, and S.-C. Li, “The choice between delta and shift operators for low-precision data representation,” *in Proc. 20th Conference of Open Innovations Association (FRUCT)*, Apr. 2017, pp. 46–52.