DESIGN OPTIMIZATION OF ANALOG ACTIVE CIRCUITS USING THE GENETIC ALGORITHM

Ioana SĂRĂCUȚ, Marius NEAG, Raul ONEȚ, Istvàn KOVÀCS
Technical University of Cluj-Napoca,
G. Baritiu Street 26-28, Cluj-Napoca, Romania
E-mail: Marius.Neag@bel.utcluj.ro

Abstract: This paper presents an optimization tool for the design of analog circuits implemented with commercially-available discrete devices. It takes into consideration the standard values available for the passive components, as well as the non-idealities of commercially-available active devices. The multi-criteria optimization is based on a genetic algorithm developed in Matlab specifically for this task, that works in conjunction with components of the OrCAD package, namely the OrCAD CIS Capture and PSpice. The effectiveness of the proposed procedure is demonstrated by a real-life example, the optimized design of a fourth-order low-pass filter implemented with Operational Amplifiers and discrete resistors and capacitors.

Key words: *analog circuit design, active filter, genetic algorithm.*

I. INTRODUCTION

Most standard procedures for analog circuit design result in non-standard values for the passive components; consequently, for a practical implementation, the calculated values are usually replaced by the nearest commercially-available ones, resulting in possible large deviations of the circuit performance from the ones targeted by the initial design. The performances can be further degraded by the second-order non-idealities of the commercially-available active devices, more difficult to take into account in the first stages of the design.

The usual solution for this problem is to adjust the component values based on simulations run iteratively, following an optimization procedure derived from the designer experience. Several CAD tools provide support for circuit optimization – for example OrCAD Circuit Optimizer and Matlab Optimization Toolbox – but most of them search for (and deliver) solutions considering a continuous range for component values and active devices parameters and do not support or provide limited support for searching a solution within a discrete range of values.

Another solution is to try out a large number of possible combinations of the available set of component values, and select the one that gives best simulation results. This brute-force approach requires large processing resources and/or simulation time; as it will be shown in this paper, tools based on *Genetic Algorithms* (GA) can be developed to perform such a search in an effective way, providing optimum solutions while using modest amounts of time and resources [1].

Other solutions have been proposed in the literature, mainly in the digital area, and very few for analog networks [2]-[7].

This paper presents a tool developed specifically for optimizing the design of active and passive analog circuits considering well defined sets of available components and devices, from the "preferred values" stated by the international standard IEC 60063 (the well-known E-series) for the resistors and capacitors to lists of pre-selected active devices. The optimization tool was implemented in Matlab; it is based on a GA tailored for this task that works in conjunction with elements of the OrCAD package for circuit design: the schematic editor and the PSpice circuit simulator.

The tool proposed here has several specific features, different from the solutions proposed previously in the literature, such as: the way the circuit is encoded for optimization, that is, the representation of the circuit in the GA; the fact that it deals with both active and passive circuits; the way the optimization criteria are defined.

Section II of the paper presents in some detail the genetic algorithm, including the description of its implementation in Matlab. The calculation of the fitness function and the choice of the optimization criteria are explained in Section III. Section IV demonstrates the effectiveness of the proposed through a practical example: the design of the channel filter for a Zero Intermediate Frequency (Zero-IF) radio receiver for IEEE 802.11n broadcasts, implemented with commercially-available Operational Amplifiers (OAs) and discrete resistors and capacitors values chosen from the standard E24 series. Main conclusions and directions for future developments are discussed in the last Section.

II. OPTIMIZED SIZING OF ANALOG CIRCUITS BASED ON A GENETIC ALGORITHM

The genetic algorithm (GA) is an optimization tool inspired by the process of natural evolution. Basically, GA works with a population whose members are called *chromosomes*; a chromosome is composed of *genes*

Manuscript received July 30, 2012; revised November 12, 2012

represented as binary strings. Each chromosome encodes a possible solution to the optimization problem. The initial population is randomly generated and evolves from generation to generation over several *evolutionary cycles* by using the specific phenomena of the evolutions: *the crossover*, *the mutation* and *the selection* [8], [9]. The best fit individual is then chosen based on its performance mark, called *fitness function*. The fitness is defined so that it accurately and consistently captures the effect of several optimization criteria.

The encoding of an individual is one of the most important aspects in genetic programming, because it affects the capability of the algorithm to result in a convenient solution. In the approach used in this paper, the GA is used to perform a search for the optimum combination of values in a database of permitted component values form within the IEC 60063 standard "E" series spanned over 4 decades of values. The total number of genes that compose a chromosome is equal to the number of circuit components to be set.

The search has to start from a feasible solution. In order to obtain it, a conventional design method is applied and the values obtained are then rounded to the nearest values within the E-series chosen by designer. The combination of values thus obtained represents *the starting solution* for the GA and each chromosome will represent a small variation around this solution. The search does not take place in a continuous range, and therefore a gene does not encode the value of the component, but the distance (in binary) between the index of the starting solution and the index of the individual's value in the chosen E-series.

The approach used in this paper follows the flowchart depicted in *Figure 1*. According to this flowchart, the procedure consists in the following steps:

1. The circuit schematic is designed and run in PSpice; at this point, the values of the components are not important, because all we need now is the circuit netlist and the simulation file which are both generated from running.
2. In Matlab, the user sets the following inputs:
   - the design specifications of the circuit;
   - the set of component values of the starting solution;
   - the available component libraries;
   - the inputs for the GA, including: the optimization parameters and their weights, the probabilities of the genetic mutation and crossover, the population size (N_{pop}) and the number of generations (N_{gen}).
3. The Matlab program creates the netlist of the circuit for the components given by the starting solution and executes the circuit running in OrCAD; the data output are saved in Matlab, the optimization parameters are calculated based on data output and then the corresponding fitness function is evaluated. This is the fitness function of the starting solution, which will be used further on.

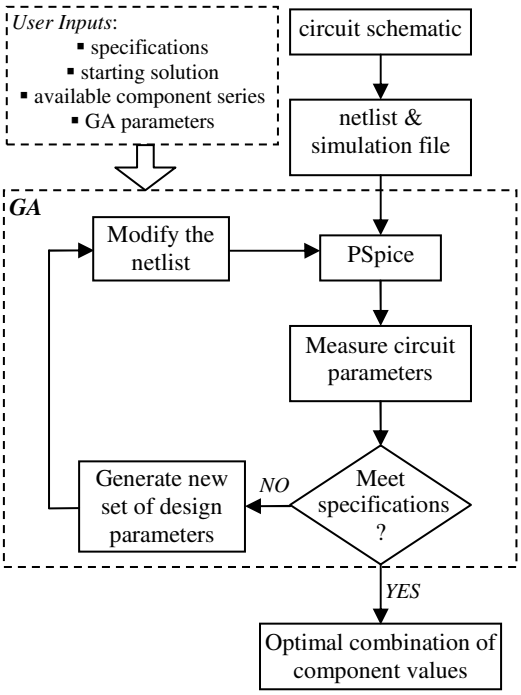

Figure 1. The flowchart of the optimization procedure.

4. The GA begins:

i) Randomly generates the initial population (N_{pop} individuals).

ii) The specific phenomena of the evolution are simulated in order to find N_{pop} more individuals. For each individual (each set of component values), Matlab creates the netlist, executes the running of the circuit in OrCAD and calculates the fitness function based on the output data.

iii) All $2N_{pop}$ individuals are sorted ascending in terms of the fitness function; the first N_{pop} individuals form the next generation.

iv) Repeat from step *ii* for N_{gen} times.

When the algorithm run concludes the lower value of the fitness function corresponds to the best solution that is the optimal combination of values and options for the circuit components.

The gene length depends on the maximum variation around the starting solution's values. At the first run of the GA, the indices of the individuals' components can be equal to the index of the starting solutions' ± maximum 3. At the second run, the starting solution is considered the best resulted from the first run and the searching is performed with a finer resolution: the maximum increment of the indices around this starting solution is ± 1 for each component value.

In the "*Measure circuit parameters*" module of the flowchart, the following performances of the circuit can be measured on the gain characteristic: the ripple in the passband, the cutoff frequency and the minimum stopband attenuation.

III. THE FITNESS FUNCTION

The fitness function establishes the rank of each individual in the population by taking into account the *optimization parameters* weighted by their corresponding

weights, set by the user. In the application of the GA envisaged in this paper – the optimization of a low-pass gain characteristic – there are 3 optimization parameters that are calculated for each individual, indicating the deviation from the ideal specifications:

(1) *the relative error of the cutoff frequency:*

$$\varepsilon_F = \frac{|F_c - F_{c0}|}{F_{c0}} \quad (1)$$

where F_c is the cutoff frequency of the individual and F_{c0} is the cutoff frequency given in specifications;

(2) *the error of the pass band ripple* ε_{Rp}; considering that the comparison is done relatively to the logarithmic characteristic (Bode plots), in the pass band this ripple ideally is 0 dB (for a low pass type), therefore the actual ripple resulted in the pass band constitutes the second error;

(3) *the error of the minimum stop band attenuation,* in terms of the minimum stop band attenuation of the individual a_{SB} and the minimum stop band attenuation given in the design specifications a_{SB0}.

$$\varepsilon_{SB} = \begin{cases} 10^6, & if\ a_{SB} \le a_{SB0} \\ \dfrac{1}{a_{SB} - a_{SB0}}, & if\ a_{SB} > a_{SB0} \end{cases} \quad (2)$$

Both values are measured for frequencies that are higher than the frequency of the first notch in the gain characteristic.

The fitness function is calculated as a weighted sum of these three deviations:

$$F_f = \varepsilon_{Rp} \cdot w_{Rp} + \varepsilon_F \cdot w_F + \varepsilon_{SB} \cdot w_{SB} \quad (3)$$

where w_{Rp}, w_F, w_{SB} are the corresponding weights. The purpose of the algorithm is to achieve the smallest possible value of the fitness function, which will correspond to the individual that is closest to meeting the design requirements.

IV. A DESIGN EXAMPLE

The effectiveness of this approach is demonstrated in this section through a practical filter example: the channel filter of a Zero-IF radio receiver for IEEE 802.11n (WLAN n) broadcasts. *Figure 2* presents the spectrum of an 40MHz-wide WLAN signal in the 5GHz band that comprises not only the wanted channel (here channel 100) but also the adjacent (channel 108) and alternate (channel 116) channels [11].

System-level analysis indicates that a 4th order inverse Chebyshev filter with the following parameters is suitable for this application:

- the low-frequency gain = 0dB;
- the maximum ripple in the pass band = 0.5 dB;
- the minimum stop-band attenuation = 20 dB;
- the -0.5 dB cutoff frequency = 20 MHz;
- the -3dB cutoff frequency = 24 MHz
- the 1 dB Input Compression Point > 5 dBVrms;
- the mean attenuation (defined below) of the adjacent and alternate channels = 20 dB;

We introduce the *mean attenuation* as a measure of the overall rejection of specific un-wanted signals located in a defined frequency region within the stop-band.

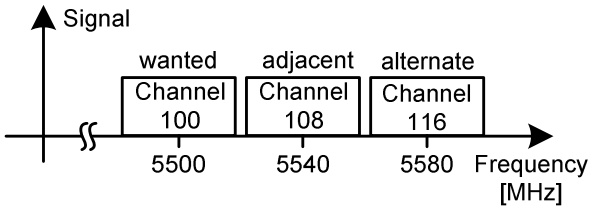

Figure 2.WLAN 40MHz channels in the 5 GHz band.

The general expression of this parameter is:

$$att_m[dB] = 20\lg|H(F_c)| - 20\lg\frac{\int_{F_c}^{F_{\max}} |H(f)|df}{F_{\max} - F_c} \quad (4)$$

where $H(f)$ is the transfer function of the filter, F_c and $F_{\max}$ are the min and max frequency values defining the region of interest within the stop-band.

The mean attenuation is particularly useful for signals with Orthogonal Frequency-Division Multiplexing (OFDM) modulation that have "brick-like" frequency spectra, for which the magnitude is almost constant within the entire channel bandwidth, as shown in *Figure 2*.

The OrCAD package comprises goal functions for PSpice, that help the user to derive directly from simulation results standard parameters such as the gain, bandwidth, slew-rate, step-response overshoot, raise and settling time [10]. However, important parameters such as *the compression point* cannot be obtained automatically. Measuring the compression point (CP) involves measuring the gain for increasingly larger levels of the input signal and detecting the level that corresponds to a decrease of the gain value by a set amount (usually 1dB). A brut-force approach is to run a (necessarily large) number of transient simulations for a complete set of input levels. An algorithm for calculating the 1 dB input compression point (1dBCP) through successive transient simulations, that minimizes the number of simulations required, is presented in *Figure 3*:

i) In order to obtain the reference (surely un-compressed) gain value one should start from a low-enough input level (point A in *Figure 3*);
ii) Then one jumps to a level close to the estimated value of the 1dBCP (point B in *Figure 3*);
iii) Next, the input level is increased iteratively; the step size can be adjusted as for a typical binary search: starting from a set value (transitions B-C and C-D in *Figure 3*) it can be halved successively (note in *Figure 3* the haled step for the transition D-E, then halved again for E - F);
iv) stop the search when results get within a set tolerance from ideal; for example, Gain(Ideal-Measured) =(0.8, 1.2)dB for 1dBCP.

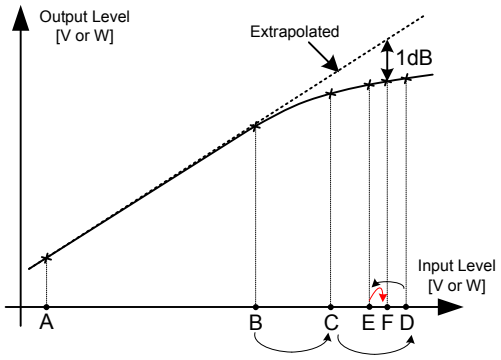

Figure 3. The algorithm for determining the Compression Point that minimizes the number of transient simulations.

The 4th order inverse Chebyshev filter was implemented by a cascade of two OA-RC biquads that realize imaginary zeros in their transfer functions [12]; the schematic is shown in *Figure 4*. The blocks denoted "-1" represent inverting voltage buffers which can be realized by cross-connecting the positive and negative inputs and outputs in fully differential implementations.

The transfer function of the 1st biquad is given by [12]:

$$H_1(s)=\frac{V_{OUT1}}{V_{IN}}=-\frac{C_a}{C_0}\cdot\frac{s^2+\dfrac{1}{C_0C_aR_0R_{a2}}}{s^2+\dfrac{1}{C_0R_{a1}}s+\dfrac{1}{C_0^2R_0R_a}} \quad (5)$$

The transfer function parameters result as follows:

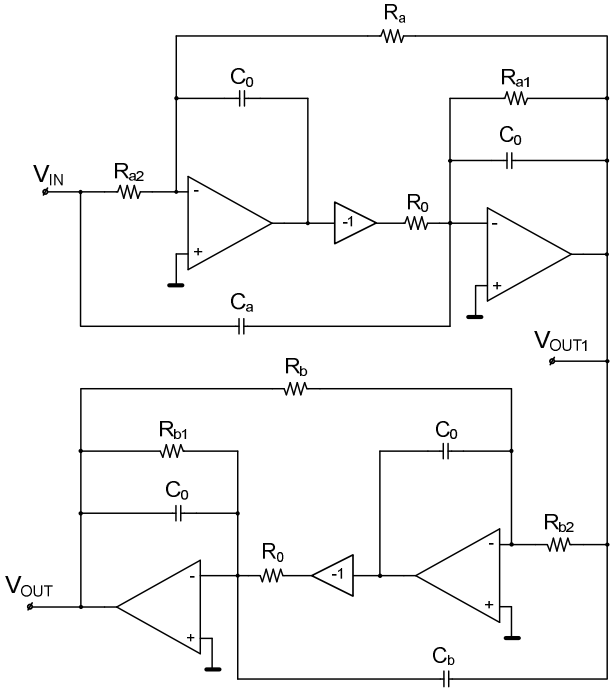

Figure 4. The schematic of the 4th order low-pass filter.

$$H_{10}=\frac{R_a}{R_{a2}}\ ;\ H_{1\infty}=\frac{C_a}{C_0}\ ;\ \frac{1}{Q_{p1}}=\frac{\sqrt{R_0R_a}}{R_{a1}}$$
$$\omega_{p1}=\frac{1}{C_0\sqrt{R_0R_a}}\ ;\ \omega_{z1}=\frac{1}{\sqrt{C_0C_aR_0R_{a2}}} \quad (6)$$

By choosing R_0 and C_0, the other components result:

$$R_a=\frac{1}{C_0^2\cdot R_0\cdot\omega_p^2}\ ;\ R_{a1}=\frac{Q_{p1}}{\omega_pC_0};$$
$$R_{a2}=\frac{R_a}{H_{10}}\ ;\ C_a=C_0\cdot H_{1\infty} \quad (7)$$

The sizing equations for the second biquad can be obtained by simply substituting the "a" index with "b".

To set a reference, we first simulated the circuit shown in *Figure 4* using the exact values resulted from the sizing equations (7) – detailed in the 2nd column of *Table 1* – and a near-ideal OA model, with a very large gain and gain-bandwidth product (GBW) and near-ideal input and output resistances. The resulting *ideal* parameters of the filter are given in 2nd column of *Table 2*.

Next, we followed the standard sizing procedure for a real-life circuit: the values of the passive components were chosen from the standard E-24 set – see the 3rd column of *Table 1*; the OPA655 OA was chosen and the model provided by the manufacturer [13] was used in simulations. The value of its GBW is 240 MHz. The resulting filter parameters – called "*before optimization*" – are given in the 3rd column of *Table 2*.

The filter magnitude characteristics for these cases are depicted in *Figure 5* with a zoom-in shown in *Figure 6*. The difference between the gain characteristics of the ideal filter (dotted line) and the filter before optimization (simple solid line) is relatively large, with a maximum value around 3dB; the filter obtained by using the standard sizing method does not satisfy the passband ripple and cut-off frequency specifications.

Finally, the passive components of the filter shown in *Figure 4* were re-sized by using the optimization tool described in Sections II and III. In general, the optimization procedure comprises several steps: for the first run the designer sets the starting point, that is the first choice for the OAs and for the standard set of values available for resistors and capacitors; if the resulting solution is not convenient, the optimization is re-run considering iteratively the following conditions:

- a finer set of values for the passive elements while keeping the same model for the OAs; (for example, E-24 instead of the initial E-12)
- return to the initial set of values for the passive but choose a better OA from the list provided by the designer; the most important OA parameters for this application are the GBW (the larger the better), the output resistance (the lower the better) and the linearity.
- increasingly finer sets of standard passive values combined with increasingly better OAs.

For the filter shown in *Figure 4* there are eight design variables available for the optimization: R_a, R_b, R_{a1}, R_{b1}, R_0, C_a, C_b and C_0. Consequently, the GA uses an 8-gene chromosome for encoding each individual. The

parameters of GA were set up as follows: length of a chromosome= 8 genes; population size = 100 individuals; number of generations = 40; the crossover rate = 80%; the mutation rate = 20%. The length of each gene was 3 bits in the first run and 2 bits in the second run – for a finer search, as explained in Section II. The OPA655 OA and the passive component values chosen following the standard sizing procedure were used as the starting point for the optimization. The fitness function defined by (3) was calculated for the weights $[w_{Rp}\ w_F\ w_{SB}] = [1\ 7\ 0.1]$.

Table 1. Components values for the three filter versions.

Components	ideal	before optimization	after optimization
R_{a1} [kΩ]	12.4	12	9.1
R_{b1} [kΩ]	2.76	2.7	2.4
$R_{a2} = R_a$ [kΩ]	7.99	8.2	7.5
$R_{b2} = R_b$ [kΩ]	4.29	4.3	3.6
R_0 [kΩ]	5	5.1	5.6
C_a [pF]	0.559	0.56	0.47
C_b [pF]	0.179	0.18	0.16
C_0 [pF]	1	1	0.91

The optimization tool found an acceptable solution while maintained the initial OA, the OPA655, and the E-24 series for selecting the passive component values from.

The new resistor and capacitor values are listed in the 4th column of *Table 1*; the resulting filter parameters are detailed in the 4th column of *Table 2*. For comparison the frequency characteristic of the optimized filter (bold line) is shown in both *Figures* 5 and *6*. The main parameters of the filter obtained for the cases analyzed here are summarized in *Table 2:* the pass-band ripple; the cutoff frequencies for 0.5dB and 3dB attenuation; the minimum attenuation in the stop band, within the [30-300 MHz] frequency range; the mean attenuation calculated using the expression (4), where F_c is the -3dB cutoff frequency, so that $20\lg\left|H\left(F_c\right)\right| = -3\ dB$, and $F_{\max} = 100\ MHz$.

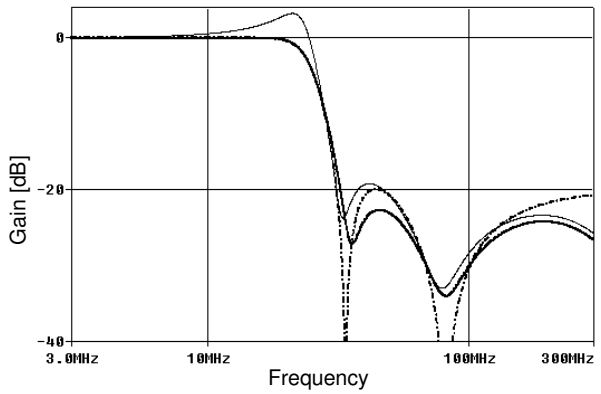

Figure 5. Gain characteristics: ideal (dotted line), before (simple solid line) and after (bold line) optimization.

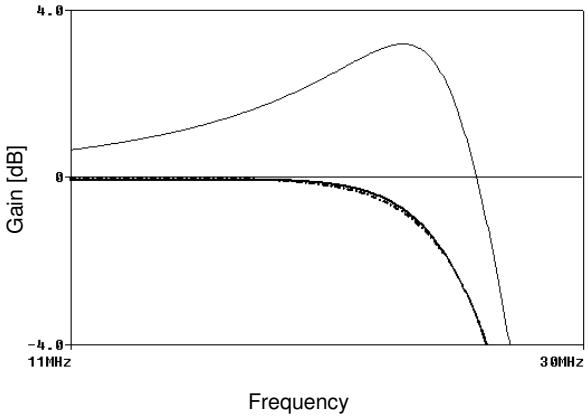

Figure 6. Zoom-in of the characteristics shown in Fig 5.

Table 2. Parameters of the three versions of the filter.

Performances	ideal	before optimization	after optimization
Ripple in the pass band [dB]	0	1.68	-0.07
-0.5dB cutoff frequency [MHz]	20	–	20.32
-3dB cutoff frequency [MHz]	24.065	23.8	24.052
Minimum attenuation [dB] in the range (30-300MHz)	20	20.7	22.72
Mean attenuation [dB] in the stop band	20.39	20.38	20

The compression point was determined using the procedure described and illustrated above in *Figure 3*. For the standard voltage follower implemented with the OPA655 supplied from a ±5V symmetric supply, driving a load of 1kΩ, the I1dBCP measured at 1 MHz was 10.5 dBVrms. The I1dBCP of the 4th order optimized filter implemented with the same OA, measured at 1 MHz, was relatively close at 5.12 dBVrms.

The effectiveness of the optimization procedure is clearly demonstrated: although the same OA is used and the component values have been changed only slightly the performances of the optimized filter are improved significantly, not only meeting all the requirements but getting very close to the performances of the ideal filter.

V. CONCLUSIONS

This paper presents an optimization tool for the design of analog circuits with a given topology, able to select the best set of passive components values and active devices form within well defined sets of available values and options. Unlike most of the existing circuit optimization tools, it can search within discrete sets of possible solutions, such as the standard series of values available commercially for passive components and lists of active devices given by the designer.

The optimization engine employs a genetic algorithm tailored for this application, which works in conjunction with elements of the OrCAD package for circuit design: the initial circuit netlist is created with OrCAD's schematic editor; PSpice provides circuit simulations for

the initial and the following netlists; some of the parameter measurements are based on OrCAD goal functions. Specific functions were developed for measurements not supported by the OrCAD package, such as the 1dB Compression Point.

The tool was implemented in Matlab; besides the GA it comprises several more functional units that ensure: the control of the simulator, the processing of the simulation results, the updating of the circuit netlist according to the optimization results.

The circuit optimization is organized as an iterative loop: the GA provides a set of values for the passive components and a selection of active devices; the circuit netlist is modified accordingly and fed to the PSpice simulator; the results are processed to derive the circuit parameters of interest – gain, bandwidth, linearity, etc. If these parameters do not meet the requirements they are conveyed to the GA which starts a new iteration.

A real-life example was presented to demonstrate the potential of the proposed tool: the optimized sizing of a 4^{th} order inverse Chebyshev filter implemented with commercially-available OAs and discrete resistors and capacitors. By comparison with the filter obtained following the standard sizing procedure the parameters of the optimized filter were improved significantly, although the optimized version employs the same OAs.

REFERENCES

[1] M. Barros, J. Guilherme, N. Horta – "Analog Circuits and Systems Optimization Based on Evolutionary Computation Techniques", *Studies in Computational Intelligence*, Springer 2010.

[2] D. H Horrocks, and M.C Spittle, " Component Value Selection for Active Filters Using Genetic Algorithms*," Proc. IEE/IEEE Workshop on Natural Algorithms in Signal Processing*, vol. 1, no. 1, pp. 13/1-13/6, Chelmsford, UK, November 1993.

[3] D. H. Horrocks, Y. M. A. Khalifa – "Genetically Derived Filter Circuits Using Preferred Value Components", *IEE Colloquium on Analogue Signal Processing*, pp. 4/1-4/5, Oxford, UK, Oct. 1994.

[4] D. H. Horrocks, Y. M. A. Khalifa – "Genetically Evolved FDNR and LeapFrog Filters Using Preferred Component Values", *Proc. European Conference on Circuit Theory and Design*, pp. 359-362, Istanbul, Turkey, Aug. 1995.

[5] T. Arslan and D. H. Horrocks – "The Design of Analogue and Digital Filters Using Genetic Algorithms", *IEE 15th SARAGA Colloquium on Digital and Analogue Filters and Filtering Systems*, 1995, pp.2/1-2/5.

[6] C. Goh, Y. Li – "GA Automated Design and Synthesis of Analog Circuits with Practical Constraints, *Proc. on the 2001 Congress of Evol..Comput.*, vol.1, pp. 170-177, May 2001, Seoul.

[7] S. Busquets-Monge *et al.* – "Design Optimization of Power Electronics Circuits using Genetic Algorithms – A Boost PFC Converter Example", *IEEE Industry Applications*, Vol 10, No. 1, February 2004.

[8] D. Dumitrescu – "*Genetic Algorithms and Evolution Strategies. Applications in Artificial Intelligence and Related Areas*" (in Romanian: "Algoritmi genetici şi strategii evolutive. Aplicaţii în inteligenţa artificială şi domenii conexe"), Editura Albastră, Cluj-Napoca, 2000.

[9] D. E. Goldberg – "*Genetic Algorithms in Search, Optimization and Machine Learning*", Adsison-Wesley, Reading – MA, 1989.

[10] A. Fazakas, M. Neag, L. Feştilă – "Improved Pspice Goal Functions Measuring Step Response Parameters", *International Conference Applied Electronics* 2005, Pilsen, Czech Republic, pp. 101-104, 2005.

[11] *IEEE Std 802.11n - 2009 Part 11*: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. (n.d.)

[12] M. Neag, L. Nedelea, M. Topa, and L. Festila – "CAD Tool for Optimized Design of High-Performance Active Filters," in *IWSSIP 2005,* International Workshop on Systems, Signals & Image Processing, pp. 389-393, 2004.

[13]Burr-Brown, "*Wideband, Unity Gain Stable, FET-Input Operational Amplifier*", OPA655 Datasheet, 1997.