Adaptive Amplifier System for Sensor Network Applications

Mónica Lovay[1], Gabriela Peretti[1,2], Eduardo Romero[1,2], Carlos Marqués[2]

[1] Grupo de Estudio en Calidad en Mecatrónica, Universidad Tecnológica Nacional, Facultad Regional Villa María, 5900 Villa María, Argentina,
gecam@frvm.utn.edu.ar
[2] Grupo de Desarrollo Electrónico e Instrumental, Universidad Nacional de Córdoba, Facultad de Matemática, Astronomía y Física, Medina Allende y Haya de Torre, 5000 Córdoba, Argentina.
marques@famaf.unc.edu.ar

Abstract. This paper presents an adaptive amplifier that is part of a sensor node in a wireless sensor network. The system presents a target gain that has to be maintained despite the presence of faults while its bandwidth must be as large as possible, without direct human intervention. The system is composed by a software-based built-in self-test scheme implemented in the node that checks all the available gains in the amplifiers, a reconfigurable amplifier and by a genetic algorithm (GA) for reconfiguring the node resources that runs in a host computer. We adopt for the node implementation a PSoC device from Cypress. The performance evaluation of the scheme presented is made by adopting two different types of fault-models in the amplifier gains. The fault simulation results show that GA finds the target gain with low error, maintains the bandwidth above the minimum tolerable bandwidth and presents a run time lower than an exhaustive search method.

Keywords: Evolvable hardware, software-based built-in self-test, genetic algorithm, adaptive amplifier system, wireless sensor networks.

1 Introduction

Wireless sensor networks are implemented with a usually large number of sensor nodes. These nodes have the ability to communicate each other by means of wireless transmission. Usually, a host computer collects data from the sensors and performs different actions depending on the particular purpose of the system. A broad range of applications have been proposed for this kind of systems such as industrial sensor networks, environmental monitoring, home automation, medical and health care among others [1].

In the above-mentioned applications, the nodes can operate under the action of a number of agents that potentially could degrade the system performance. If the application is critical, the system can require characteristics of safe operation, adaptation to a changing environment or ability for compensating degradations in its own circuitry. For achieving this purpose, two related characteristics are necessary: fault detection and self-adaptation to a changing environment.

A typical wireless sensor node comprises sensor processing and communication units. In this context, microcontrollers are god candidates for implementing part of a node because they offer some benefits. These include low cost and power consumption, ability to perform data processing tasks in the node and usually, powerful communication interfaces. In addition, modern microcontrollers (μCs) offer a wide pool of configurable digital and analog sections that enhance the chip ability for adapting to a broad range of applications.

The programmable analog sections in μCs offer an alternative to traditional fault tolerant schemes because the reconfigurable nature of these devices enables runtime correction [2]. Additionally, although reconfiguration not always guarantees that a complete functionality can be restored, it allows maintaining the system operation with slight degradation [3].

Particularly, evolvable hardware (EHW) is a methodology that combines reconfigurable hardware with evolutionary algorithms with the aim of adapting a system to changing environments or providing fault recovery. In this methodology, the designer establishes performance goals and a GA searches the possible hardware configurations for reaching them [2], [3]. Relevant work in the area of fault tolerance and fault recovery of electronic circuits can be found in [4-7].

EHW usually require a test strategy for detecting the presence of hardware faults in order to establish that it is necessary a reconfiguration. Regarding the test of configurable analog circuits, in [8], [9], is presented an on-line testing strategy for continuous-time field programmable analog arrays (FPAAs). In [10-12], techniques such as oscillation-based test and transient analysis method have been successfully applied to FPAAs for testing interconnection resources and basic building blocks.

In this paper, we present an adaptive amplifier that is part of a sensor node in a wireless sensor network. The system presents a target gain that has to be maintained despite the presence of faults while its bandwidth must be as large as possible, without direct human intervention. We employ for implementing the GA a host computer, which is commonly used in sensor networks. The system is composed by a software-based built-in self-test (SW-BIST) scheme (implemented in the node) that checks all the available gains in the amplifiers, a reconfigurable amplifier and by a GA for reconfiguring the node resources that runs in a host computer. We adopt for the node implementation a PSoC device from Cypress.

2 System Description

PSoC device is a programmable system-on-chip platform with an on-chip processor core [13]. It includes configurable blocks of analog circuits, programmable interconnect and configurable IO in a low-cost chip. Analog functions in the device are organized as groups of general-purpose analog blocks that can be configured into user-determined functions. The control for these blocks is register-based and can be programmed through design tools or reprogrammed by the user at run-time. Some of the available configurations for the analog arrays are up to 14 bits analog to digital converters (ADC), up to 9 bits digital to analog converters (DAC), programmable gain amplifiers (PGA), programmable filters and comparators.

The PGA user module implements an operational amplifier based non-inverting amplifier with user-programmable gain (Fig. 1). This amplifier has high input impedance, wide bandwidth, and selectable reference. There are 33 programmable values for the PGA gain, ranging from 0.062 to 48 [14].

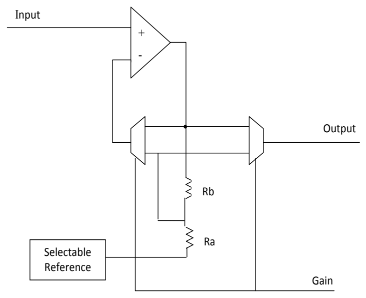

Fig. 1. Programmable gain amplifier available in the PSoC® device (simplified diagram).

We assume that each node of the wireless sensor network requires an adaptive amplifier. Particularly, the adaptive amplifier addressed in this work employs three PGAs. Fig. 2 shows the amplifier system present in one node of the sensor network and the interaction with the host computer. The three-amplifier chain (PGA1, PGA2 and PGA3) is configured in the PSoC CY8C27443-24PXI.

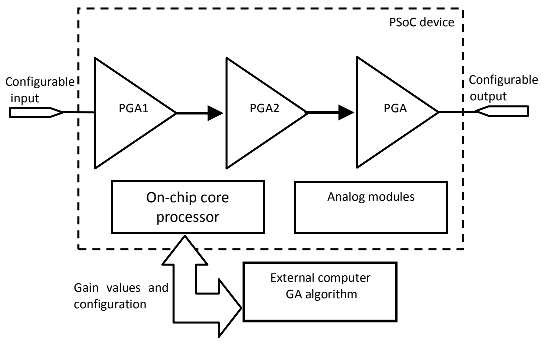

Fig. 2. Amplifier system diagram, normal mode.

A measurement process (to be described in the next section) tests the gain of each amplifier during the dead times of the system. The node transmits the test data to the host for its evaluation. If the test finds a degradation in the gains, then establishes that is necessary a system reconfiguration.

The reconfiguration involves the use of a GA running in the host computer. GA evolves the gain values of the three amplifiers with the goal of maintaining the system overall gain within specifications and the bandwidth as large as possible. The evolved values of gain are loaded into the hardware for continuing the normal operation. In this work, it is assumed that the tolerable error in the global gain is lower or equal to the error in the gain values reported in the PGA datasheet, in our case ±5% of the gain nominal value [14]. This value could be redefined according to the application needs.

3 PGA Gain Test

The test of each PGA gain is a necessary step for achieving adaptation to a changing environment or for tolerating faults. For this process, we use the processor core, on-chip analog resources and the dynamic reconfiguration characteristic of the PSoC device. In this way, this testing approach virtually eliminates the need for additional test-specific hardware. Based on this characteristic, this test proposal is contextualized as an embedded SW-BIST method [15-17].

When the gain test starts, a first PGA (PGA1 in Fig. 3) is disconnected from the chain. A nine-bit DAC connects to the PGA1 input a DC signal while a twelve-bit ADC converts the PGA1 output to a digital value. During the PGA1 gain test, the remaining amplifiers are not active. The on-chip processor calculates the gain value and establishes the communication with the external computer where is running the GA. The test process continues with the measurement (by means of reconfiguration) of the gain of the remaining amplifiers in a sequential way.

Once the test is finished, the test data are transmitted to the computer. If degradation is detected, the GA evolves the system. After this process, the processor receives the new values of gain for the amplifier system.

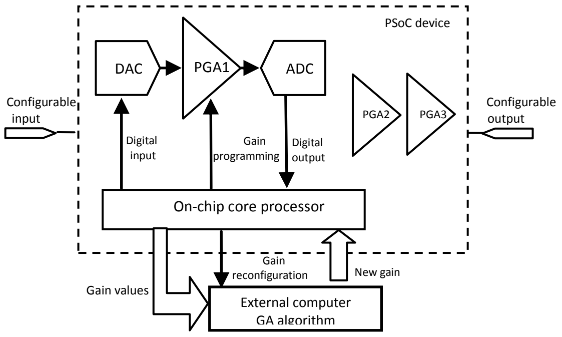

Fig. 3. PGA1 gain test diagram.

4 EHW Overview

In this work, we adopt an extrinsic EHW methodology [3]. As previously stated, GA must evolve the gain values of the three amplifiers with the goal of maintaining the system overall gain within specifications and the bandwidth as large as possible.

For our case, we consider that the most important objective is to maintain the overall gain within specifications. This fact allows using the so-called apriori methods. One of these is the "ε-Constraint Method" that transforms a multi-objective problem into another with a single objective. It performs the optimization with respect to an objective and transforms the others into restrictions [18, 19]. The use of this method allows employing traditional GAs, as the described in [2-3, 20].

GA randomly generates (with uniform probability) an initial population of individuals that are possible solutions to the problem. Each solution in the population is a string of bits, also called chromosomes, or genotypes. For every evolutionary step, known as a generation, the individuals in the current population are evaluated according to some predefined quality criterion called the fitness function. To form a new population (the next generation), individuals are selected according to their fitness; high-fitness individuals present better chances to appear ("survive") in the next generation, while low-fitness ones are more likely to disappear.

Constrained optimization problems require the use of some methods in order to apply constraints in the problem solving. The penalty is a technique for dealing with the constraints. By using this technique, the solutions violating the restrictions are modified in their fitness values (based on the violation degree) in order to decrease their chance of being selected.

Genetically inspired operators, crossover and mutation generate the individuals for the next generation. The crossover operator selects two individuals, called parents, and exchanges parts of their information (the string of bits) to form two new individuals, called offspring. In its simplest form, bits from one parent are exchanged for bits from the other parent, creating two offspring. If the two parents do not undergo the crossover operation, they are copied unchanged to the new pool of individuals.

The mutation operator is applied to the new pool of individuals produced after the application of crossover. This operator prevents premature convergence to local optima by flipping bits (of individuals) at random with some probability. After mutation, a new generation of individuals is produced. This new generation goes through the process described above, from the fitness evaluation to the mutation step. The cycle repeats until a stop criterion is met, such as a maximum number of generations is reached or a desired solution is found.

5 Genetic Algorithm Parameters

The GA has to find the three PGA gain values (G_1 for PGA1, G_2 for PGA2 and G_3 for PGA3) that reach the condition:

$$\mathrm{Min}(\ |A_{tar} - G_1 \ . \ G_2 \ . \ G_3|\ ) \quad \text{subject to } BW \geq \varepsilon. \tag{1}$$

In (1), A_{tar} is the target gain and ε is the minimum tolerable bandwidth (BW) of the system.

The bandwidth of the adaptive amplifier is found as the real positive solution to the eq. (2) [21]. For formulating this equation, it is assumed that each PGA is modeled as a first order system, as reported by the vendor.

$$\prod_{k=1}^{3}\left[1+\left(\frac{BW}{pk}\right)^{2}\right]=2\cdot \tag{2}$$

In (2), *pk* is the k-th pole of each PGA. Its value is calculated as:

$$pk = \mathrm{GBWP}/G_k, \quad \text{if } G_k \geq 1; \qquad pk = \mathrm{GBWP}, \quad \text{if } G_k < 1. \tag{3}$$

In (3), GBWP is the gain bandwidth product reported by the vendor and G_k is the gain of the k-th amplifier.

For simplifying the operation of the GA, the equation of the fitness function (f) is formulated for obtaining a maximum [2]:

$$f = B - |A_{tar} - G_1 \,.\, G_2 \,.\, G_3|, \tag{4}$$

where B is a constant added for avoiding negative numbers.

For the PGA gain values, it is used a simple binary codification. The creation of the population in the first cycle of the algorithm is made by using uniform initialization. The population size is 30, and the number of generations is 30. The size of the population is chosen according to the guidelines of [20] and ensures that the probability of finding a binary value 1 or a binary value 0 at each position in the chromosome exceeds 99.9%. The fitness of each individual is calculated using (4). The individuals that present an overall gain within specification and a bandwidth greater than or equal to ε are modified in their fitness. The individuals with higher bandwidth are assigned with higher fitness. This change is an increase proportional to the difference between the current bandwidth of the individual and ε:

$$\mathrm{NewFitness} = \mathrm{CurrentFitness} + \mathrm{CurrentFitness}\,.[(\mathrm{CurrentBandwidth} - \varepsilon)/\,\varepsilon]. \tag{5}$$

On the other hand, the individuals with BW below ε, even if they present a gain within specification, are penalized by diminishing his fitness in a proportional way to the difference between ε and its current bandwidth:

$$\mathrm{NewFitness} = \mathrm{CurrentFitness} - \mathrm{CurrentFitness}.[(\varepsilon - \mathrm{CurrentBandwidth})/\,\varepsilon]. \tag{6}$$

The selection of the individuals for the crossover is performed through the method of the rotating roulette. The probability of being selected for an individual is proportional to its fitness. The probability of crossover and mutation are both fixed in 0.5. These values are chosen using previous experimental guidelines [4, 20].

6 Experimental Results

6.1 PGA Gain Test

The gain test was performed in four different PSoC® devices at two different temperatures (25°C and 50°C). All the measurement errors showed values close to those reported in the PGA datasheet [14]. Additionally, it was found that the relative error between the programmed gain (expected value) and the measured gain was below 5% in the range of gain values [0.062-8]. We observed wide deviations from the expected values (outside the above-mentioned gain range), when chip-to-chip or intra chip (between PGAs of the same chip) comparisons are made and an erratic behavior at different temperatures. For this reason, we limit the range of possible values of gain to this range for lowering the error when applying the EHW technique. Fig. 4 shows one of the PGA gain test results [22].

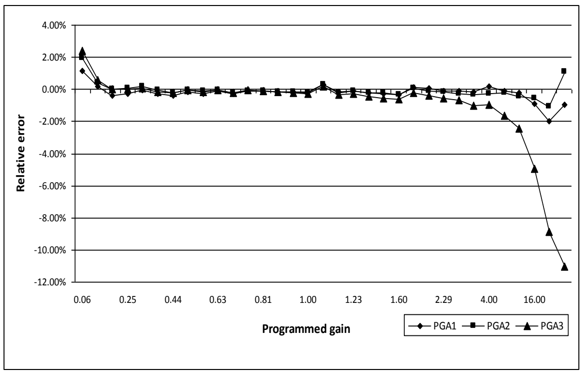

Fig. 4. Errors in the test gain process. Fault-free operation at 25°.

6.2 Fault Free Operation

We use in the GA the gains values obtained during the gain test process and consider that they are all available. We propose three different values for the target gain (A_{tar}): 2, 8 and 15 with the aim of evaluating the ability of GA for finding an acceptable solution in different scenarios. We set the value of the minimum -3dB BW (ε) in 4E+06 rad/s (636 kHz), as the minimum acceptable value for future applications.

The distribution of the obtained results can be observed in the dispersion diagram depicted in Fig. 5. This figure shows the relative error of the gain versus the bandwidth for the three target gains for several runs of GA. Each point is a solution to the optimization problem changing the seed for the random generation of the first population (see Section 5). In this way, we evaluate the GA performance with different initial populations. From the figure, the three target gains present relative errors in the range [-4.60%, 4.92%]. The lowest BW obtained for all the evaluated gains is 4.23E+6 rad/s, above the required.

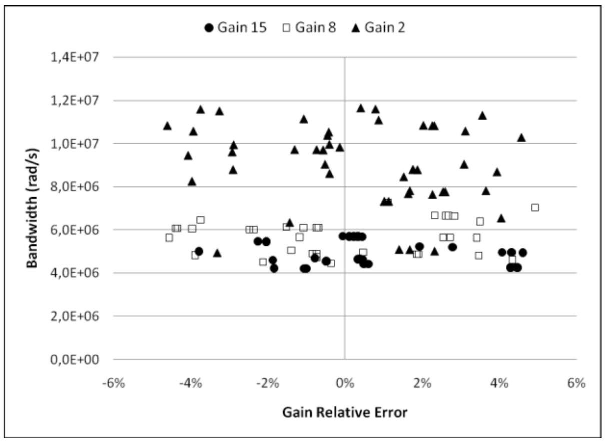

Fig. 5. Gain relative error versus bandwidth. Fault-free operation.

Table 1 shows a characterization of the gain relative error for the three target gain values. We adopt the median as a measurement of central tendency because the data distribution is not normal. We also present the maximum, minimum and range values as a measurement of dispersion. We found that the error range for all the cases is below the required (+/- 5%). Additionally, the median of the relative error is close to zero for all target gains.

Table 1. Gain relative error characterization under normal condition.

Target Gain	Median (%)	Minimum error (%)	Maximum error (%)	Error range (%)
15	0,34	-3,78	4,60	8,38
8	-0,69	-4,55	4,92	9,47
2	1,01	-4,60	4,56	9,16

6.3 Operation under Fault Condition

The performance of the scheme presented here is evaluated by means of fault injection. To this end, it is necessary to define a fault model.

If the PGA is well designed, the operational amplifier can present wide deviations in its functional parameters without effects in its closed loop performance. Consequently, we consider that the main cause of PGA gain faults comes from faults or degradations in the resistances that establish the gain. In each PGA, we consider two different types of faults in the gain determined by the resistances Ra y Rb (Fig.1). The first one is a catastrophic fault that assume that is not possible to establish one gain value. The second fault is a deviation in the gain values.

Because PGAs are embedded in the PSOC devices, it is impossible to inject faults directly in the hardware. For this reason, we adopt a different approach. For injecting a catastrophic fault, we eliminate from the search space used by the GA the gain value that it is assumed as faulty. For injecting deviation-faults, all the gains values in the search space are deviated by the same amount.

Fig. 6 depicts the results obtained under catastrophic fault condition for the PGA1, which presents the worst performance. The figure shows the relative error of the gain versus the bandwidth for the three target gains. Each point is a solution to the optimization problem when is removed a gain value in the PGA.

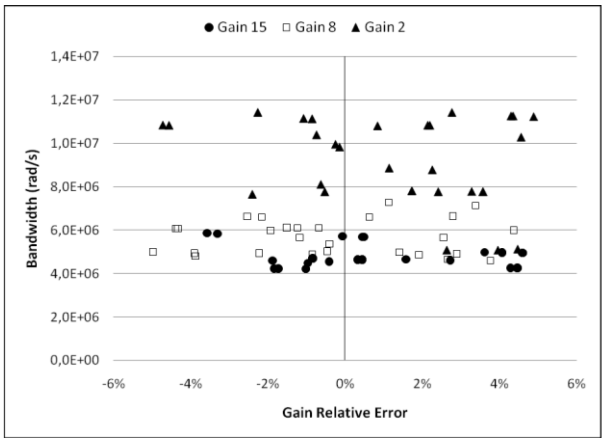

Fig. 6. Gain relative error versus bandwidth. Catastrophic fault operation in PGA1.

Table 2 summarizes the results under catastrophic fault conditions in the three PGAs. In this table, we grouped the results by gain value. Comparing the gain error results obtained from normal operation (Table 1) and catastrophic fault operation (Table 2), the faulty system presents as a worst case an increase of 0.53% in the error range for gain 2. The median of the relative error remains almost constant for gain 15, while for the other two gains presents variations, suggesting in these two cases that the error distribution changes between the normal and faulty operation. In all the experiments, the GA is capable of reaching the target gain, with errors for all the gains in the range [-4.98%, 4.89%]. The lowest BW obtained for all the simulated conditions is 4.23E+6 rad/s, above the required.

Table 2. Gain relative error characterization under catastrophic fault condition.

Target Gain	Median (%)	Minimum error (%)	Maximum error (%)	Error range (%)
15	0,33	-3,89	4,60	8,49
8	-0,47	-4,98	4,36	9,33
2	0,82	-4,80	4,89	9,69

For deviation faults, we consider that PGAs present a deviation in their gain values in a percentage of their nominal values, ± 10%, ±20%, ±30%, ±40% and ±50%. Fig. 7 shows the deviation-fault simulation results for the PGA1, which presents the worst performance. The figure depicts the relative errors in the target gains versus the BW obtained for each deviation value in the gain. From the simulation results, it is observed that the GA is able to reach the target gain with errors for all the gains in the range [-4.91%, 4.85%] and BW greater than 4.10E+06 rad/s. Once again, in the worst case, the obtained BW is above the required.

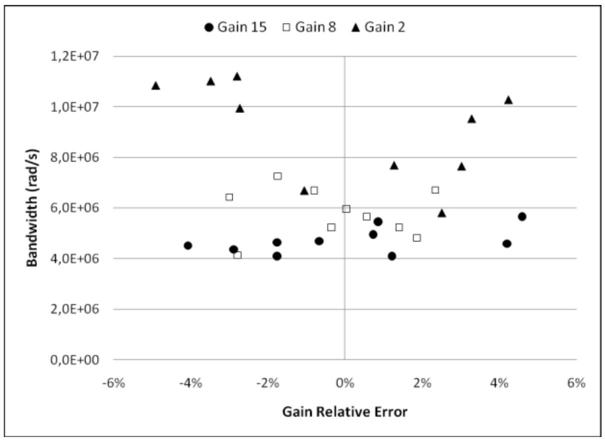

Fig. 7. Gain relative error versus bandwidth. Deviation fault operation in PGA1.

Table 3 summarizes the effects of deviation faults in the three PGAs. Comparing the relative error under normal (Table 1) and deviation fault conditions (Table 3), the faulty system presents an increase of 0.60% for gain 2, and a slight diminution for the other two gains, despite the presence of relatively high deviation faults.

Table 3. Gain relative error characterization under deviation fault condition.

Target Gain	Median (%)	Minimum error (%)	Maximum error (%)	Error range (%)
15	-1,18	-4,76	4,59	9,35
8	-0,79	-4,86	4,54	9,40
2	-1,06	-4,91	4,85	9,76

7 Comparison with Exhaustive Search Method

For a better characterization of the efficiency of our genetic algorithm, we compare it with Exhaustive Search Method (ESM). This method consists of systematically enumerating all possible candidates for the solution and checking whether each candidate satisfies the problem statement [23]. We chose to perform this comparison due to the relatively low number of possible gain values achievable by the overall system (900 gain values). Under this condition, this brute force method looks like a serious competitor for heuristic methods. Particularly, it could appear that this method is considerably faster than GA.

We performed the comparison using two parameters: number of objective function evaluations and run time. Table 4 shows the obtained results. The ESM must perform 27000 objective function evaluations (eq.(4)) to find the best solution for normal operation and deviation fault operation. For catastrophic fault operation, this method must perform 26100 evaluations. By other way, the GA performs at most 900 evaluations (population size x number of generations).

As can be observed from the results reported in Table 4, the number of objective functions evaluations has an important impact on the run time. The run time of GA shown is the median of the run time obtained in the worst case (for gain 15). In the worst condition (deviation fault), GA is 36.76 times faster than ESM. In the best condition (catastrophic fault), GA is 59.07 times faster than ESM.

Table 4. Performance comparison between ESM and GA.

Method	Run Time (sec)			Objective Function Evaluations		
	Normal condition	Fault condition		Normal condition	Fault condition	
		Catastrophic	Deviation		Catastrophic	Deviation
GA	0,048	0,043	0,072	900	900	900
ESM	2,647	2,540	2,647	27000	26100	27000

On the other hand, we compare the bandwidth corresponding to the gain values obtained by the GA with the bandwidths obtained by using ESM under normal operation and faulty operation. The bandwidths for the GA are close to optimal bandwidths (obtained with the ESM) with the median between 16.61% and 26.96% lower. However, we emphasize that these results were obtained in considerably less time and with fewer objective function evaluations.

8 Conclusions and Future Work

We presented an adaptive amplifier implemented with programmable gain amplifiers in a PSOC device that is part of a sensor node in a wireless sensor network. The system is composed by a SW-BIST scheme that check all the available gains in the amplifier and by a GA for reconfiguring the chip resources that runs in a host computer. The GA presented is robust for the types of faults addressed in our evaluation. The fault simulation results show that the system maintains the overall gain and the bandwidth within specifications, despite the presence of catastrophic and deviation faults. In addition, its run time is considerably lower than of an exhaustive search method.

References

1. Flammini A., Ferrari P., Marioli D., Sisinni E., Taroni A., "Wired and wireless sensor networks for industrial applications". Microelectronics Journal, V. 40, pp.1322--1336 (2009)
2. Goldberg D., Genetic Algorithm. Search, optimization and machine learning, United States: Addison-Wesley (1989)
3. Salem Zebulum R., Pacheco M., Vellasco M., Evolutionary electronics: automatic design of electronic circuits and systems by genetic algorithms, United States: CRC Press (2002)
4. Hereford J., “Fault-tolerant sensor systems using evolvable hardware”, IEEE Trans. Instrum. Meas, vol. 55, pp. 846--853 (2006)

5. Emmert J., Stroud C., Abramovici M., “Online fault tolerance for FPGA logic blocks”, IEEE Trans. Very Large Scale Integr. VLSI Syst., vol. 15, pp. 216--226 (2007)
6. Haddow P. C., Hartmann M., Djupdal A., “Addressing the metric challenge: evolved versus traditional fault tolerant circuits”, in Proc. Second NASA/ESA Conference on Adaptive Hardware and Systems, Edinburgh, pp. 431--438 (2007)
7. Ji Q., Wang Y., Xie M., Cui J., “Research on fault-tolerance of analog circuits based on evolvable hardware”, in Proc. 7th international conference on Evolvable systems: from biology to hardware, Wuhan, China, pp. 100--08 (2007)
8. Wang, H., Kilkarni, S., Tragoudas, S.: On-line testing field programmable analog arrays circuits. In: Proc. International Test Conference, pp. 1340--1348 (2004)
9. Laknaur, A., Wang, H.: A Methodology to Perform Online Self-Testing for Field-Programmable Analog Arrays Circuits. IEEE Transactions Inst. and Meas., Vol. 54, Nº 5, 1751--1760 (2005)
10. Balen, T., Andrade, Jr., Azais, F., Lubaszewski, M., Renovell, M.: Testing the configurable analog blocks of field programmable analog arrays. In: Proc Int. Test Conf., pp 893--902 (2004)
11. Pereira, G., Andrade, Jr., Balen, T., Lubaszewski, M., Azais, F., Renovell, M.: Testing the interconnect networks and I/O resources of field programmable analog arrays. In: Proc. 23rd IEEE VLSI Test Symposium, pp. 389--394 (2005)
12. Balen, T., Andrade, Jr., Azais, F., Lubaszewski, M., Renovell, M.: Applying the Oscillation Test Strategy to FPAA’s Configurable Analog Blocks. Journal of Electronic Testing, Vol. 21, pp. 135--146 (2005)
13. PSoC® Programmable system-on-chip technical reference manual. United States: Cypress Semiconductor Corporation (2008)
14. Programmable gain amplifier data sheet, United States: Cypress Semiconductor Corporation, (2009)
15. Krstic, A., Wei-Cheng L., Kwang-Ting C., Chen, L., Dey, S., “Embedded software-based self-test for programmable core-based designs”, IEEE Des. Test Comput., vol. 54, pp. 18--27 (2002)
16. Kesh A., “Software-based BIST for analog to digital converters in SoC”, in IEEE International Design and Test Workshop, Cairo, pp. 189--192 (2007)
17. Psarakis M., Gizopoulus D., Sanchez E., Sonza Reorda M., “Microprocessor software-based self-testing”, IEEE Des. Test Comput., vol. 27, pp. 4--8 (2010)
18. Collette Y., Siarry P., “Multiobjective optimization: principles and case studies”, Springer (2003)
19. Branke J., Deb K., Miettinen K., “Multiobjective optimization: interactive and evolutionary approaches”, Springer (2008)
20. Reeves, Rowe J., Genetic algorithms: principles and perspective. A guide to GA theory, United States: Kluwer Academic Publishers (2002)
21. Centurelli F., Monsurro P., Trifiletti, A., “Power-constrained bandwidth optimization in cascaded open-loop amplifiers”, ECCTD 2007. 18th European Conference on Circuit Theory and Design, pp. 65--654 (2007)
22. Lovay M., Arregui A., Gonella J., Peretti G., Romero E., Lubaszewski M., “Fault Tolerant Amplifier System Using Evolvable Hardware”, Proceedings of the Argentine School of Micro-Nanoelectronics, Technology and Applications, pp. 50--55 (2010)
23. Price K., Storn R., Lampinen J., “Differential evolution: a practical approach to global optimization”, Springer (2005)