An Analog VLSI Multilayer Perceptron and its Application Towards Built-In Self-Test in Analog Circuits

Dzmitry Maliuk*, Haralampos-G. Stratigopoulos‡ and Yiorgos Makris*
* Electrical Engineering Department, Yale University, 10 Hillhouse Ave, New Haven, CT 06520-8267, USA
‡TIMA Laboratory (CNRS-INP Grenoble-UJF), 46 Av. Félix Viallet, 38031 Grenoble, France

***Abstract*— A proof-of-concept hardware neural network for the purpose of analog built-in self-test is presented. The network is reconfigurable into any one-hidden-layer topology within the constraints of the number of inputs and neurons. Analog operation domain of synapses and neurons in conjunction with the digital weight storage allow fast computational time, low power consumption and fast training cycle. The network is trained in the chip-in-the-loop fashion with the simulated annealing-based parallel weight perturbation training algorithm. Its effectiveness in learning how to separate nominal from faulty circuits is investigated on two case studies: a Butterworth filter and an operational amplifier. The results are compared to those of the software neural networks of equivalent topologies and limitations concerning the practical applicability are discussed.**

I. Introduction

Built-in self-test (BIST) is the ability of an integrated circuit (IC) to examine its own functional health, in order to detect and report faults that may jeopardize the reliability of the application wherein it is deployed. BIST can be used for off-line test, aiming to simplify test resources, speed up test time, facilitate fault diagnosis, and obtain valuable information for yield enhancement in future IC generations. However, the key advantage of BIST is that it can also be performed on-line in the field of operation and, thus, target malfunctions due to environmental impact and wear. Therefore, BIST is vital to ICs that are deployed in safety-critical applications (e.g. avionics, medicine, nuclear reactors), sensitive environments (e.g. space operations), and remote-controlled systems.

Successful application of BIST in the analog domain has long posed a great challenge. Traditionally, analog circuits are tested by measuring directly all performance parameters and subsequently comparing them to the specifications. This typically involves complex automatic test equipment (ATE) and demands lengthy test times. Evidently, migrating ATE capabilities onto the IC for the purpose of BIST is impractical.

A promising solution relies on simple on-chip sensors to extract informative measurements. These measurements can be mapped to the performance parameters using the alternate test paradigm [1], [2], [3]. Alternatively, instead of predicting individual performance parameters, one can directly predict the outcome of the standard specification test. This approach narrows down to a binary classifier, which produces a pass/fail decision indicating the health status of the circuit [4], [5], [6].

The above mappings can be performed externally on the ATE or using software on the baseband DSP processor [7]. The latter choice points to a true stand-alone BIST. However, a DSP processor might not always be accessible or even present on the chip and might not have the necessary power to learn the mapping through a training phase. In addition, the interface of the analog circuit to the DSP processor would require A/D converters. Finally, low power is a serious concern when BIST is to be used for concurrent test or frequent on-line tests in battery operated systems. A more aggressive approach would be to build the mapping on hardware. The classification approach is particularly suitable for on-chip integration since it can be implemented using hardware neural networks that accept analog inputs, provide a binary response, and operate in the sub-threshold regime.

The BIST architecture that we envision is illustrated in Fig. 1. When BIST is enabled, the circuit is driven by the test stimulus and a measurement pattern is recorded by the available on-chip sensors. The measurement pattern is next processed through the neural classifier and is classified as a valid or invalid code-word pointing to a functional or faulty operation, respectively. The classifier learns to perform this mapping in a training phase that is carried out before BIST is enabled. This phase employs a representative data set of the circuit which is stored off-chip.

The focus of this work is the classifier, which is a generic BIST component independent of the circuit under test. In particular, the aim is to demonstrate the feasibility of learning on-chip complex classification problems related to BIST. To this end, we present the design of a classifier as a stand-alone

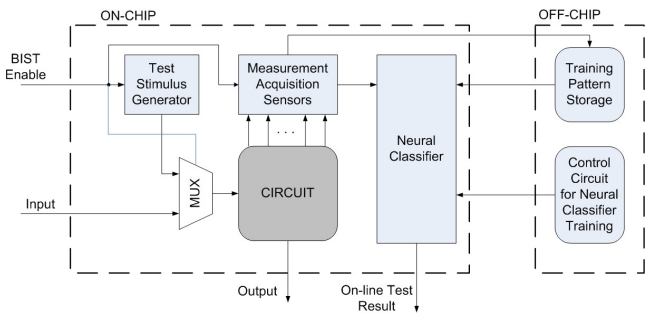

Fig. 1. Proposed BIST architecture.

978-1-4244-7723-4/$26.00

IC chip, we discuss its learning strategy, and we demonstrate its efficiency on two classification problems stemming from a Butterworth low-pass filter and an operational amplifier. Our primary objective was to design a reconfigurable and flexible architecture so as to study various hardware classifier models and gain insight into their performance and limitations compared to their software counterparts. This study serves as a proof-of-concept of on-chip learning and classification. The issue of cost-effective implementation requires further investigation and will be the subject of future research work.

II. Neural Network Design

A. Background

In designing an analog neural network one has to consider a number of important factors. Appropriate connectionist topologies, training algorithms, long-term weight storage are among the most crucial. In addition, there are design constraints imposed by a BIST application itself, such as small area overhead, limited pin number, negligible interference, etc. Furthermore, one has to consider implications of the technology in which a network is to be implemented. Digital CMOS processes, which are becoming more popular for analog/RF circuits, are plagued by process variation, mismatch, noise, environmental factors, etc. There is a large body of literature discussing these factors and various architectural considerations [8], [9], [10], [11]. Further discussion, however, is outside the scope of this work. The network presented herein was designed with two key objectives in mind: reconfigurability to explore various network models for a given application and simplicity of training. The first was achieved by utilizing control bits to program the connectivity of the network, while the second by using digital weight storage in local RAM cells.

B. System Architecture

The chosen model for the neural classifier is a multi-layer perceptron (MLP), which is a feed-forward network consisting of several interconnected layers of neurons. Each node in this network has a simple mathematical model shown in Fig. 2. A synapse can be considered as a multiplier of an input signal value by the stored weight value. A neuron sums the output values of the connected synapses and passes the resulting sum through a nonlinear activation function. Thus, the entire network requires only two types of circuits, efficient implementation of which is essential for large networks.

Fig. 3 illustrates a network architecture that can be reconfigured into any one-hidden-layer topology within the given number of inputs and neurons. The network consists of a matrix of synaptic blocks (S) and neurons (N). The synapses represent mixed-signal devices in the sense that they conduct all computations in analog form, however, their weights are implemented as digital words stored in a local RAM memory. Multiplexers before each synapse are used to program the source of its input: either the primary input (for a hidden layer) or the output of other neurons (for an output layer). The results of synapse multiplication are summed and fed to the corresponding neuron, which performs a squashing function

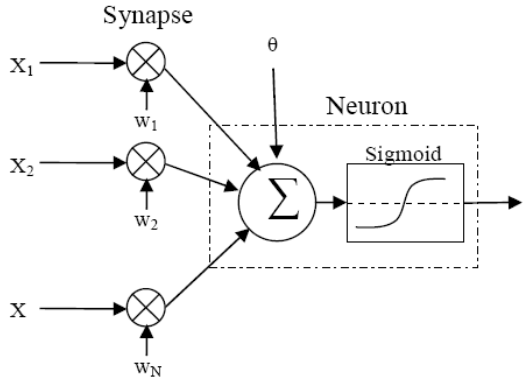

Fig. 2. Neuron and synapse models.

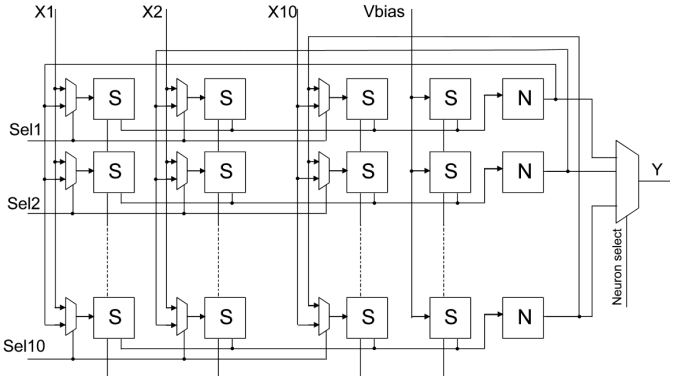

Fig. 3. Reconfigurable network architecture.

and produces an output either to the next layer or the primary output. The architecture is very modular and can easily be expanded to any number of neurons and inputs within the available silicon area. The output muliplexer is introduced to reduce the number of pins and ADCs. The signal encoding takes different forms: the outputs of the neurons are voltages, while the outputs of the synapses are currents. In addition, all the signals are in differential form increasing the input range and improving noise resilience.

C. The Synapse Circuit

The basic function of a synapse is multiplication. Linear multiplication, as dictated by the mathematical model, is area expensive in analog ICs. As a result, it is often the case that much simpler circuits exhibiting only approximate multiplication behavior are preferred. The effect of nonlinearity in synapse circuits is addressed in a number of resources, with the solutions ranging from specially tailored backpropagation algorithms [8] to the training algorithms independent of synapse and neuron transfer characteristics [12].

The synapse circuit chosen for this design is a simple multiplying DAC [13], which represents a differential pair with programmable tail current (Fig. 4). A differential input voltage produces a differential output current which is collected on the summing nodes common to all synapses connected to one neuron. The core of the circuit is a differential pair $N10 - N11$ performing a two-quadrant multiplication, while the four

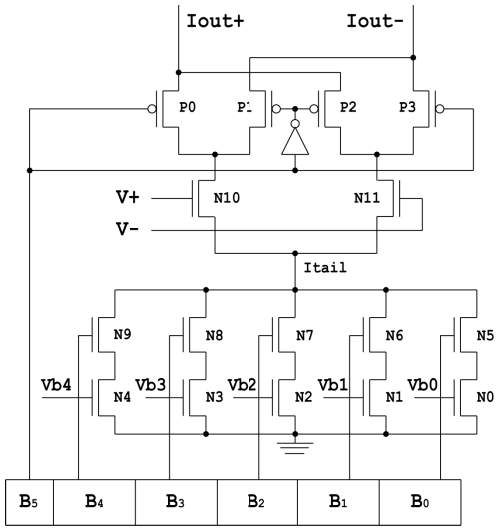

Fig. 4. Synapse circuit schematic.

switching transistors $P0 - P3$ steer the current between the two summing nodes for a four-quadrant multiplication. The tail current is digitally controlled by the five switch transistors $N5 - N9$ connecting the corresponding binary weighted current sources $N0 - N4$ to the tail node. Thus, the tail current as a function of a digital weight word (bits $B4 - B0$) can be represented by

$$I_{tail} = \sum_{i=0}^{4} B_i \cdot I_i = \sum_{i=0}^{4} B_i \cdot I_{bias} \cdot 2^{i-4} = I_{bias} \cdot W, \quad (1)$$

where B_i are the bits of a weight word, I_i is the current corresponding to the i-th bit, I_{bias} is the external biasing current, and W is the weight value. The biasing voltages Vb_i for the current sources of all synapses are supplied by a single biasing circuit shown in Fig. 5. The external biasing current I_{bias} sets the MSB current component, while the other currents are generated internally using the ratioed current mirrors.

The input-output relation between the variables of the synapse circuit depends on the region of operation of the differential pair. In the above threshold region this relation using the first order transistor models takes the following form

$$\Delta I_{out} = K_N \Delta V_{in} \sqrt{\frac{I_{bias} W}{K_N} - (\Delta V_{in})^2}. \quad (2)$$

Here, ΔV_{in} is the differential input voltage and K_N is the transconductance coefficient. Linear multiplication is only valid for a narrow range of differential input voltages. As can be seen from the above equation, this range depends on both the tail current and the transconductance coefficient. Since synapses account for the major part of power consumption, the tail current was kept minimum, but enough for the differential pair to operate just above threshold. The external biasing current during actual experimentation was about 1.5 μA producing the LSB current of about 90 nA, which corresponds to the weak inversion. In view of the small operating currents the relatively wide input range (about 800 mV for maximum weight) was achieved by selecting transistors with long channels (14 μm). A proper choice of

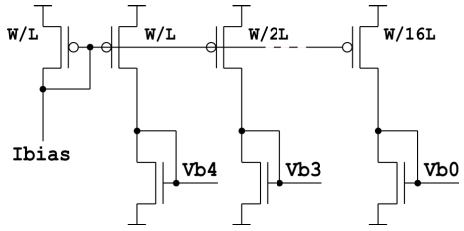

Fig. 5. Current sources control circuit.

the common mode input voltage alleviates the problem of the large gate-source voltage drops and keeps the current sources in saturation. Other transistors were kept at minimum size.

D. The Neuron Circuit

The main function of a neuron circuit is to convert the sum of differential currents from its synapses into a differential voltage. There are two issues that need to be taken into account when designing this circuit. First, if the output voltage is propagated to the next layer, it should be compatible with the input requirements of the synapses, i.e. have high common mode. Second, the circuit should handle relatively large dynamic range of input currents. While the useful information is contained in the difference, the common mode current may vary significantly depending on the number of connected synapses, as well as on their weight values. In the current design, the common mode current ranges from 90 nA (one synapse, minimum weight value) to 30 μA (10 synapses, maximum weight values).

A circuit satisfying these requirements is shown in Fig. 6(a). The central part of the circuit is responsible for common mode cancellation by subtracting the input currents from each other and producing a positive difference. The output currents of the transistors $N7$ and $N0$ can be expressed as $max(0, (I_{in}^{+} - I_{in}^{-}))$ and $max(0, (I_{in}^{-} - I_{in}^{+}))$ respectively. Thus, only one of the transistors can sink non-zero current at a time. The second stage is a simple current-to-voltage converter composed of two p-channel MOSFETs. It can be shown that when the transistors are identical such circuit exhibits a linear to the first degree characteristic of the following form

$$V = V_{dd} - \frac{I}{2K_P(V_{dd} - 2V_{TP})}, \quad (3)$$

where K_P is the transconductance coefficient, V_{TP} is the threshold voltage, and V_{dd} is the supply voltage. The circuit also provides a limiting function when the input current exceeds the internal current flowing through the circuit, thus introducing nonlinearity to the neuron characteristic. Note from the formula above that the slope of the characteristic depends on the K_P, which is set at the design stage by specifying transistor sizes. Finally, the output of the converter is shifted upwards to meet the requirements of the high common mode input voltage for the synapses in the following layer. This level shifter is a simple source follower circuit where the amount of shift is controlled by the V_{bias}. A shift of 1 V is used in this design. Fig. 6(b) shows the simulated transfer characteristic of the entire circuit and represents the activation function of the neuron.

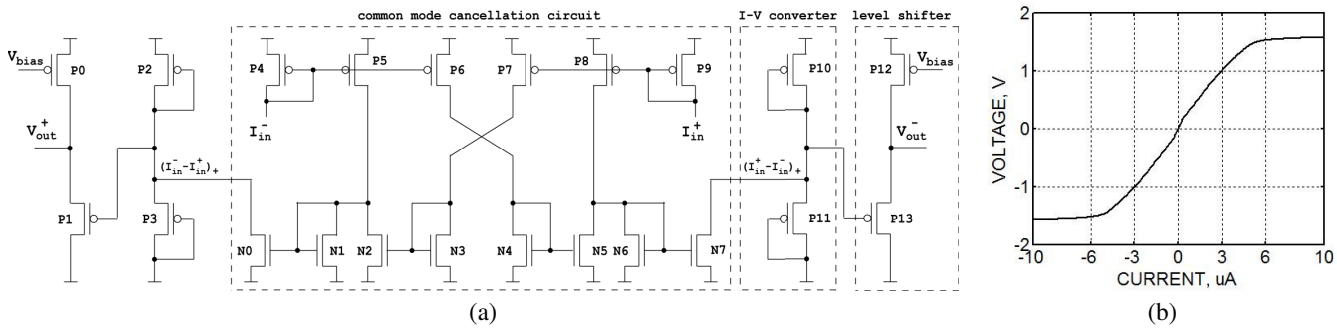

Fig. 6. (a) Neuron circuit schematic; (b) Neuron characteristic.

E. Training setup

The choice of a learning algorithm is critical for analog neural networks and depends on what paradigm of training is pursued, i.e. on-chip, off-chip or chip-in-the-loop. The latter approach is particularly attractive since it requires little or no on-chip support resulting in a compact solution. The chip is trained by a computer program which uses the network's response to guide the search in the weight space. The gradient of the error surface cannot be calculated, since the model of the network is unknown in addition to the weights being non-continuous. Instead, it can be measured by perturbing a set of weights and observing a change in the error function. Such weight perturbation-based algorithms are well suited for analog implementations, because they make no assumption about the network model, i.e. the synapse and neuron characteristics.

For this design we employed a popular algorithm called parallel stochastic weight perturbation [12]. In this algorithm, all weights are perturbed simultaneously by a random vector. Then the mean squared error is evaluated on the entire training set. If the error decreases, the new vector of weights is accepted; otherwise, it is discarded. This algorithm, however, suffers from high likelihood of convergence to a local minimum. Thus, training may need to be performed several times before a good solution is found. To decrease the probability of being stuck in a local minimum, this algorithm has been augmented with the simulated annealing technique, which is known to be efficient in avoiding local minima since it allows the state of the network to move "uphill". The main difference from the original algorithm consists in its ability to accept weight changes resulting in an increase of the error, however, with a certain probability. This probability depends on the magnitude of the error change and the "temperature" of the system T, i.e. $p \simeq \exp(-\Delta E/T)$. Higher temperatures at the initial stages favor the exploration of the whole search space. A cooling schedule is used to adjust the temperature and magnitude of weight perturbations as the training progresses. In general, this training approach applied to the hardware network has shown very similar performance in comparison to software neural networks, as will be shown later.

III. Chip implementation and experimental results

The chip has been fabricated using a 0.5 μm digital CMOS process available through MOSIS. Fig. 7 shows a photograph of the chip and Table I summarizes its key features. The

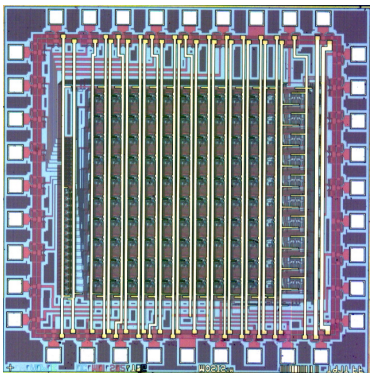

Fig. 7. Chip photograph.

training algorithm runs in MATLAB and communicates with the chip via a layer of two PCB boards. The top layer is a custom-built PCB board that houses the chip, as well as two DACs, an ADC, and biasing circuits. The bottom layer is a commercial FPGA board that was used for communication with the PC and for programming the on-chip memory, the DACs, and the ADC.

The chip is tested on two case studies: a Butterworth low-pass filter and an operational amplifier. The main objective is to train the chip to discriminate good from faulty circuit instances based on a set of low-cost measurements. The data sets are borrowed from [4], where the software-based approach has proved to perform well. In brief, a set of 14 low-cost measurements is collected from about 2000 instances of each circuit, including ac, dc, impulse response, Fourier transformation of the power supply current, and power supply ramping. The course of the experiment is as follows:

1) To explore the trade-offs between the dimensionality of the problem and classification accuracy, we perform a feature selection algorithm to determine the best measurement subsets for each cardinality. A combination of a genetic algorithm and an ontogenic neural network have been used for this purpose [6].
2) The data set is randomly split into training and validation sets in proportion 3/1. The training set is used to train the chip and the validation set is used to report the classification error. Due to the stochastic nature of random splitting and training, these steps are repeated 10 times and we report the mean error rate and the standard

TABLE I

CHIP KEY FEATURES

Implementation method	mixed-signal
Network topology	reconfigurable MLP
IC process	0.5 μm CMOS
Core area	$1 \times 1.2\ mm^2$
Neurons	10
Synapses	100
Weight resolution	6 bit
Response time	$< 1\ ms$
Power supply	3.3 V
Max. current per synapse	3 μA

deviation. Different network models are explored by varying the number of hidden neurons.

3) For comparison purposes, we repeat the same experiment using an exact software version of the MLP network in the Matlab Neural Networks toolbox. Training, however, is performed using the superior gradient-based resilient backpropagation algorithm.

The ability of the chip to generalize on the validation set is shown in Fig. 8. The error is plotted versus the best measurement subsets of cardinalities from 2 to 7. The figure shows the error for 3 different network topologies that consist of a single layer of 2, 4, and 8 hidden neurons (hn), respectively. It can be seen that the software classifiers obtain smaller errors than their hardware counterparts; however, the differences are small. Notice that there are about 500 circuits in the validation sets, which means that 0.2% error translates to approximately 1 circuit being misclassified. There are mainly two reasons for this difference in error. First, the software classifiers are trained using a superior learning algorithm. Second, the weights in the hardware classifiers have finite precision and limited dynamic range. During the experiments it was often observed that the weights saturated to their maximum value. Despite these two disadvantages, the experiment illustrates that the hardware classifier achieves similar performance compared to an ideal software classifier and is capable of learning to discriminate good from faulty circuits.

For the Butterworth data set, the errors of both the hardware and software classifiers tend to decrease as the number of hidden neurons and the dimensionality increase. For the operational amplifier data set, the performance is best for medium-size measurement subsets, which is a phenomenon called curse of dimensionality [4]. Another observation that can be made is that the performance of the software classifier improves drastically from 2 to 4 hidden neurons. In general, it can be concluded that satisfactory results can be accomplished with networks of moderate complexity.

The standard deviation of the classification errors is shown in Fig. 9. It can be observed that the variance of the error is one order of magnitude lower than its mean value. Moreover, it is independent of the network topology, the dimensionality of the measurement subset, and the type of classifier (e.g. software or hardware). The low variance of the error of the hardware classifier indicates good stability properties of the hardware training algorithm.

An interesting observation concerning the behavior of the hardware classifier can be made from Fig. 10 which shows

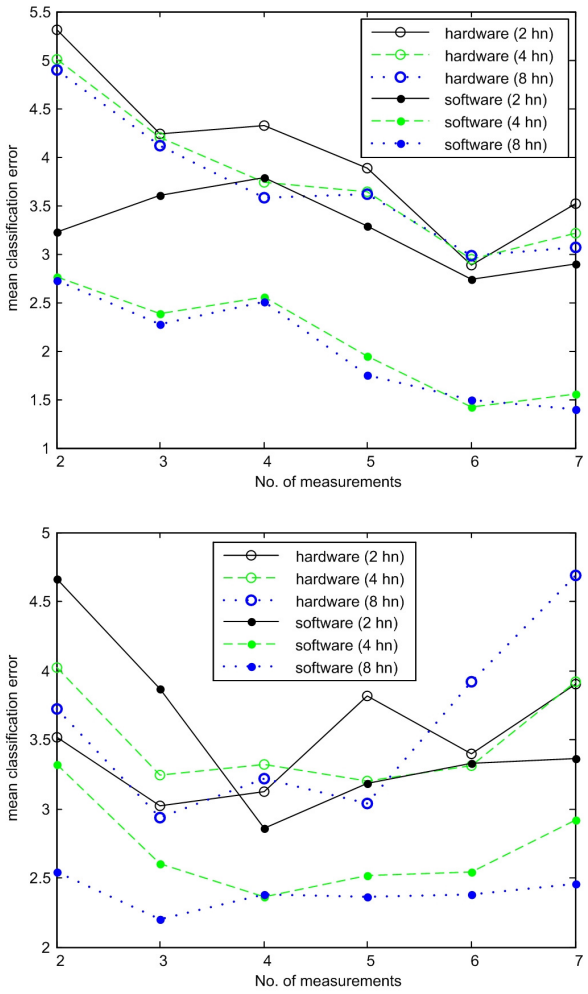

Fig. 8. Mean classification error for the Butterworth filter (upper) and operational amplifier (bottom).

the classification rates on the training and validation sets for both classifiers with respect to the number of measurements. As expected, the software network over-fits the training data resulting in a gap between the training and validation errors. However, the problem of over-fitting seems to be avoided by the hardware classifier – the error on the training set can be used as an indication of its performance.

IV. CONCLUSION

This paper proposed a BIST architecture for analog circuits and focused on its central component, namely the neural classifier. The role of this neural classifier is to map a set of simple on-chip measurements to a single-bit decision, which indicates whether the performances of the circuit comply to the specifications or not. To this end, we described in detail a reconfigurable neural classifier chip which we designed and fabricated using a 0.5 μm digital CMOS process available through MOSIS. The chip was tested using data sets from two analog circuits, namely a Butterworth filter and an operational amplifier. Our experiments showed that the chip's ability to separate good and faulty devices is comparable to its software counterpart. The aim of the paper was to provide a proof-

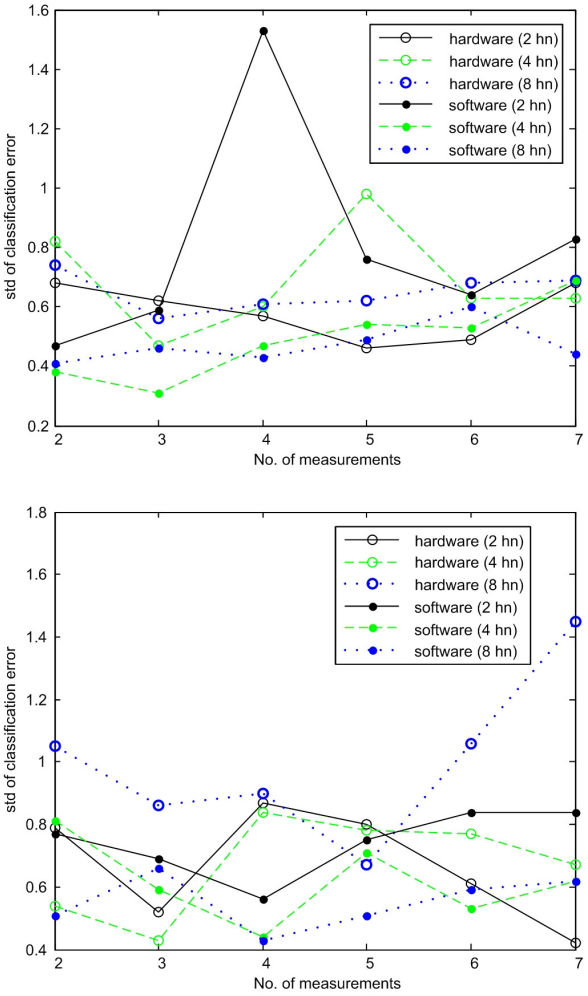

Fig. 9. Standard deviation of the classification error for the Butterworth filter (upper) and operational amplifier (bottom).

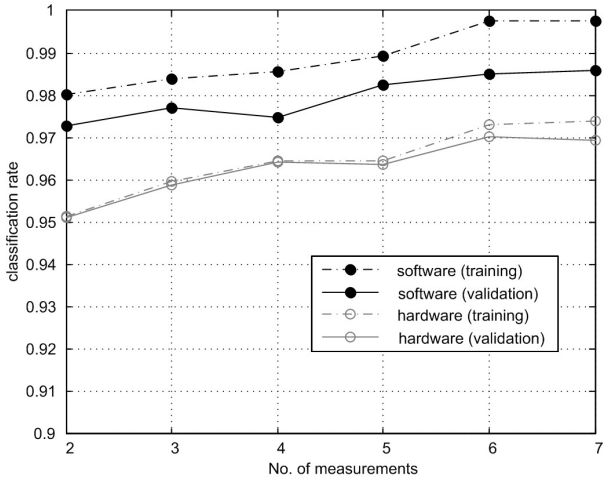

Fig. 10. Butterworth filter training results for 8 hidden neurons.

of-concept of on-chip learning capabilities in the context of analog BIST. Further work will focus on cost-effective implementations. In particular, we plan to investigate the use of floating-gate synapses, which constitute an elegant solution for long-term nonvolatile memory storage, as well as precise weight updates and local learning [14]. Furthermore, we plan to explore the trade-off between the classification error and the overhead of the neural network by moderating its complexity and by using guard-bands to separate high confidence regions of the decision space via a safety layer [6].

ACKNOWLEDGMENTS

This research has been carried out with the support of the National Science Foundation (NSF ECS-0622081 and CCF-0702522), the Semiconductor Research Corporation (SRC-1836.029) and a Marie Curie International Reintegration Grant (MIRG-CT-2007-209653).

REFERENCES

[1] S. S. Akbay and A. Chatterjee, "Feature extraction based built-in alternate test of RF components using a noise reference," in *IEEE VLSI Test Symposium*, 2004, pp. 273 – 278.

[2] S. S. Akbay and A. Chatterjee, "Built-in test of RF components using mapped feature extraction sensors," in *IEEE VLSI Test Symposium*, 2005, pp. 243–248.

[3] D. Han and A. Chatterjee, "Robust built-in test of RF ICs using envelope detectors," in *IEEE Asian Test Symposium*, 2005, pp. 2–7.

[4] H.-G. D. Stratigopoulos and Y. Makris, "Non-linear decision boundaries for testing analog circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 11, pp. 1760–1773, 2005.

[5] H.-G. D. Stratigopoulos and Y. Makris, "Bridging the accuracy of functional and machine-learning-based mixed-signal testing," in *IEEE VLSI Test Symposium*, 2006, pp. 406–411.

[6] H.-G. D. Stratigopoulos and Y. Makris, "Error moderation in low-cost machine learning-based analog/RF testing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 2, pp. 339–351, 2008.

[7] D. Han, B. S. Kim, and A. Chatterjee, "DSP-driven self-tuning of RF circuits for process-induced performance variability," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 2, pp. 305–314, 2010.

[8] J. Lont and W. Guggenbuhl, "Analog CMOS implementation of a multilayer perceptron with nonlinear synapses," *IEEE Transactions on Neural Networks*, vol. 3, no. 3, pp. 457–465, 1992.

[9] M. Milev and M. Hristov, "Analog implementation of ANN with inherent quadratic nonlinearity of the synapses," *IEEE Transactions on Neural Networks*, vol. 14, no. 5, pp. 1187–1200, 2003.

[10] S. Tam and M. Holler, "An electrically trainable artificial neural network (ETANN) with 10240 floating gate synapses," in *Procedings of International Joint Conference on Neural Networks*, 1989, pp. 191–196.

[11] A. Montalvo, "Toward a general-purpose analog VLSI neural network with on-chip learning," *IEEE Transactions on Neural Networks*, vol. 8, no. 2, pp. 413–423, 1997.

[12] M. Jabri and B. Flower, "Weight perturbation: an optimal architecture and learning technique for analog VLSI feedforward and recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 3, no. 1, pp. 154–157, 1992.

[13] V. F. Koosh and R. M. Goodmanr, "Analog VLSI neural network with digital perturbative learning," *IEEE Transactions on Circuits and Systems - II*, vol. 49, no. 5, pp. 359–368, 2002.

[14] S. C. Liu, J. Kramer, G. Indiveri, T. Delbrück, and R. Douglas, *Analog VLSI: Circuits and Principles*, MIT Press, 2002.