# An Analog Non-Volatile Neural Network Platform for Prototyping RF BIST Solutions

Dzmitry Maliuk* and Yiorgos Makris†
*Electrical Engineering Department, Yale University, New Haven, CT, 06520-8267
†Electrical Engineering Department, The University of Texas at Dallas, Richardson, TX, 75080-3021

***Abstract*— We introduce an analog non-volatile neural network chip which serves as an experimentation platform for prototyping custom classifiers for on-chip integration towards fully stand-alone built-in self-test (BIST) solutions for RF circuits. Our chip consists of a reconfigurable array of synapses and neurons operating below threshold and featuring sub-$\mu$W power consumption. The synapse circuits employ dynamic weight storage for fast bidirectional weight updates during training. The learned weights are then copied onto analog floating gate (FG) memory for permanent storage. The chip architecture supports two learning models: a multilayer perceptron and an ontogenic neural network. A benchmark XOR task is first employed to evaluate the overall learning capability of our chip. The BIST-related effectiveness is then evaluated on two case studies: the detection of parametric and catastrophic faults in an LNA and an RF front-end circuits, respectively.**

## I. Introduction

Machine learning-based testing of RF circuits has demonstrated the use of software non-linear classifiers to predict pass/fail test labels from low-cost measurements [1]. This approach has paved the way for developing stand-alone BIST solutions for RF circuits by integrating stimuli generators, low-cost sensors and a hardware implementation of non-linear classifiers alongside the circuit-under-test (CUT), as shown in the architecture of Fig. 1. In this architecture, the responses of the on-chip sensors to the stimuli generators, usually provided in the form of DC voltages, are presented to the neural classifier which produces a binary output indicating whether the CUT passes or fails its specifications. The accuracy of such prediction depends both on the ability of the neural classifier to learn the underlying mapping (i.e. its learning ability) and on the separability of classes as such (i.e. sensor measurement quality). The latter has been addressed in recent studies showing great promise in predicting performances of CUTs with minimum overhead [2]. Therefore, this work focuses on the former and investigates a hardware implementation of two popular classifier models — a multilayer perceptron (MLP) and an ontogenic neural network (ONN). Prior research has demonstrated high efficiency of these models running in software to predict circuit health from low-cost measurements [1]. However, the processing resources required to run the software models, such as an external computer or a built-in digital signal processor, are not always available to a stand-alone integrated circuit (IC), thus calling for a custom hardware network that can be integrated on-chip.

978-3-9815370-2-4/DATE14/©2014 EDAA

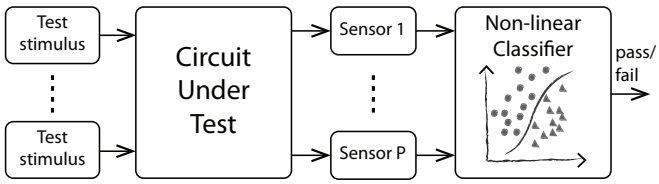


Fig. 1. Components of the RF BIST architecture. The CUT is excited by test stimuli. Multiple sensors collect simple measurements which are processed by the non-linear classifier that produces a pass/fail test result.

In applications such as BIST the additional power and area overhead incurred by test circuits is crucial. Accordingly, we selected an analog neural network implementation for its superior power efficiency during run time, compact size and the possibility of permanent weight storage using analog FG memory. Indeed, when high precision is not required, analog computation can be as much as 1000x more energy efficient than digital [3]. In addition, analog circuits can be directly interfaced with sensor outputs, thereby eliminating the use of analog-to-digital converters. Finally, the use of the FG technology in standard CMOS offers efficient non-volatile storage of weight values with high accuracy [4].

Our previous work in this field focused on demonstrating a proof-of-concept of learning in mixed-signal neural networks [5] and using an emulation model of an experimental floating gate chip [6]. In this work, we present an analog non-volatile neural network experimentation platform fabricated in a 0.35-$\mu$m CMOS process from TSMC, as well as its silicon learning results. The platform features a large number of components and serves as a prototyping tool for identifying appropriate classifier models and their parameters for a given classification problem. Thereby, the final classifier circuitry to be integrated on chip is optimized to deliver high classification accuracy with minimum area/power overhead.

The remainder of the paper is structured as follows. Section II introduces two classifier models and outlines their training algorithms. Section III describes the implementation details including the overall architecture, the weight storage mechanism, as well as the synapse and neuron circuits. Section IV-A evaluates the chip's learning capability on a XOR task. In Section IV-B, we present a first case study where the neural classifier is trained to distinguish between functional and faulty RF LNA chips based on amplitude detector measurements. In the second case study, presented in Section IV-C, the neural

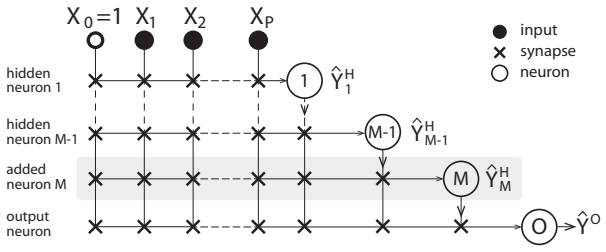


Fig. 2. The ontogenic neural network topology. The hidden neurons receive connections from both the network inputs $X_i$ and the outputs of the previous hidden neurons $\widehat{Y}_j^H$. The bottom neuron serves as a network output $\widehat{Y}^O$.

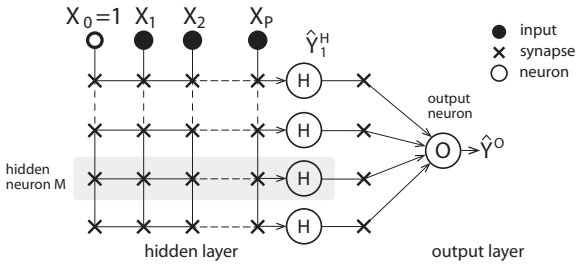


Fig. 3. The multilayer perceptron topology with one hidden layer and one output neuron.

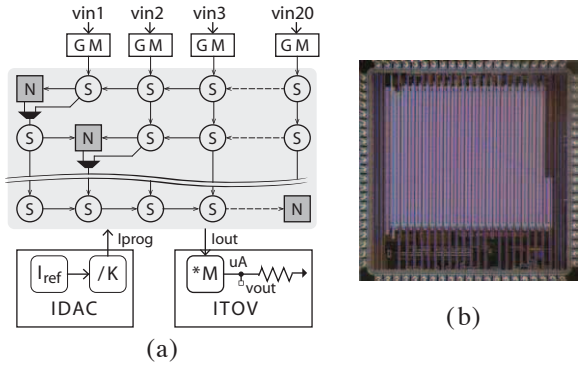


Fig. 4. (a) System architecture of the neural network chip. The reconfigurable array of synapses (S) and neurons (N) is shown in the shaded box. (b) Die photograph of the chip implemented in a 0.35-$\mu$m CMOS and measuring 3×3 mm$^2$.

classifier is trained to identify defects in RF front-end chip instances based on the readings from various on-chip sensors. Finally, Section V concludes the paper.

## II. Overview of Neural Classifiers

### A. Ontogenic Neural Network

Fig. 2 illustrates a block diagram of the ONN learning model. A decision boundary is constructed by successively adding hidden neurons (HN); each hidden neuron augments the feature space of the original inputs with the intention of making the derived space linearly separable. This strategy is guided by the cascade-correlation algorithm [7], which repeats the following steps for each added neuron. Suppose that our current stage has $M-1$ hidden neurons, as shown in Fig. 2. Let $\widehat{Y}_i^H$ be the output of the $i$-th hidden neuron and $\widehat{Y}_i^O$ be the output of the network when it has $i$ hidden neurons. The $M$-th hidden neuron is added at the bottom so that it sees the primary inputs $X_0, ..., X_P$ as well as all the outputs of the preceding neurons $\widehat{Y}_1^H, ..., \widehat{Y}_{M-1}^H$. Next, we train this neuron to maximize the correlation between its output $\widehat{Y}_M^H$ and the training error of the previous stage $\widehat{E}_{M-1} = \sum (Y_T - \widehat{Y}_{M-1}^O)^2$, where $Y_T$ represents the target class labels and the summation is done over the entire training set. Once the correlation is maximized, the weights of this neuron become permanent and the output layer is retrained to minimize the error on the training set, i.e. $\widehat{E}_M = \sum (Y_T - \widehat{Y}_M^O)^2$. Note that in each step, only the weights of the neuron being added undergo modification, followed by the weights of the output neuron, while the other weights are kept unchanged. This feature greatly simplifies the gradient estimation by the hardware and leads to stable performance even for large-sized topologies. Hidden neurons are added until a stopping criterion is reached, which in our case is the classification error on a validation set. The correlation maximization and the error minimization are done by the resilient back propagation algorithm (iRPROP+) [8], which can be efficiently customized for hardware networks.

### B. Multilayer Perceptron

Unlike the ONN model, a multilayer perceptron learns its boundary by adjusting synaptic weights only, i.e. its topology is fixed during training. A typical network consists of two layers of neurons (Fig. 3). The first layer (a.k.a. hidden layer) receives connections from the primary inputs $X_1, ..., X_P$ and a constant $X_0 = 1$. The output layer consists of a single neuron (for binary classification) receiving its connections from the outputs of the hidden layer. The number of hidden neurons is the only parameter that needs to be selected prior to training. Once the topology is fixed, the training is performed once and for the entire network. Similarly to ONN, error minimization on the training set is achieved using a hardware customized resilient back propagation (iRPROP+).

## III. Chip Design

### A. System Description

Fig. 4 shows a block diagram of the neural network chip. A 30×20 array of synapses and neurons is arranged so that the neurons are aligned along the main diagonal of the upper matrix and along the right edge for the bottom part. Global connectivity is programmable by means of multiplexors inserted between rows. The core operates in the analog domain with weights and signals represented by differential currents. A single weight value requires two current sources for differential current storage. A current source is implemented as a current storage cell (CSC) circuit that combines two modes of operation: *dynamic*, for fast bidirectional weight updates, and *non-volatile*, for long-term storage of learned weights. The dynamic mode is engaged during training, when the weight values undergo multiple updates. Upon completion of training, the learned weights are copied onto the FG transistors for permanent storage. The peripheral circuits provide support for fast programing and interfacing with the external world. The GM blocks convert voltage-encoded input signals (sensor readings) into balanced differential currents required by the core.

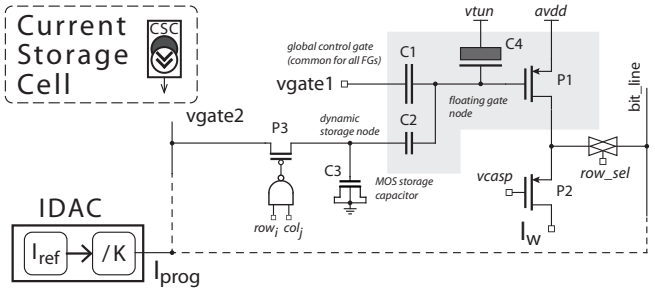


Fig. 5. Schematic of the current storage cell along with the dynamic programming loop. The sizes of key components are as follows: P1 = 2×2 $\mu$m$^2$, P2 = 2×1 $\mu$m$^2$, P3 = 0.4×0.35 $\mu$m$^2$; C1 = 40 fF, C2 = 15 fF, C3 = 1.35 pF, C4 = 0.6 fF.

The digitally-controlled current source IDAC generates target currents from an on-chip reference for dynamic programming of the CSCs. Finally, the current-to-voltage converter ITOV facilitates the reading of internal currents by converting them to voltages that can be sampled by an external ADC. Each of the blocks undergoes extensive characterization to provide a reading/sourcing accuracy of at least 8 bits.

### B. Weight Storage

The principle of weight storage is illustrated in Fig. 5. We use a multiple-input FG transistor (FGT) P1 to store the drain current $I_w$ representing one of the weight value components. The drain current is modulated by the voltage on the FG node, which is itself determined by the FG node charge and the voltages on two control gates. The global voltage *vgate1* of the first control gate is shared among all FGTs, while *vgate2* is stored locally in the dynamic sample-and-hold (S/H) circuit which consists of the switch transistor P3 and the MOS capacitor C3. The low-coupling capacitor C2 makes $I_w$ much less sensitive to charge leakage and other parasitic effects of the S/H circuit. The tunneling capacitor C4 is implemented as a minimum size PMOS transistor with its source, drain and well terminals connected to *vtun*. Hot-electron injection is used to add electrons to the FG, thus, lowering its voltage and increasing the drain current. Conversely, Fowler-Nordheim (FN) tunneling is used to remove electrons from the FG. The tunneling is used for global erase only, while the injection allows us to individually program drain currents of each CSC with high accuracy (>8 bits).

### C. Synapse and Neuron Circuits

The synapse circuit, illustrated in Fig. 6, implements a four-quadrant multiplication of a differential input current $\{I_{in}^+, I_{in}^-\}$ by a differential weight current $\{I_w^+, I_w^-\}$. The circuit features two CSC cells for differential weight component storage and a six-transistor core P1-P6. The neuron circuit, illustrated in Fig. 7, implements a nonlinear activation function of the sum of the outputs of the connected synapses. This nonlinear transformation is completed in two stages. The first stage, represented by the bottom part of the circuit, controls the slope of the activation function. The slope is adjusted by programming the $I_{gain}$ current, which is stored in a local CSC.

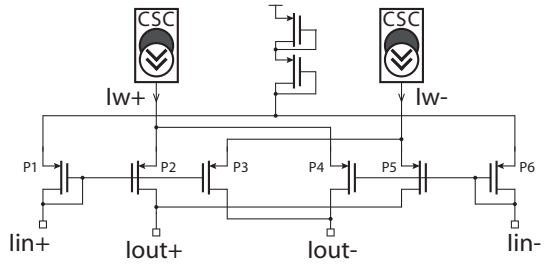


Fig. 6. Schematic of the synapse circuit (P1 = ... = P6 = 4×2 $\mu$m$^2$).

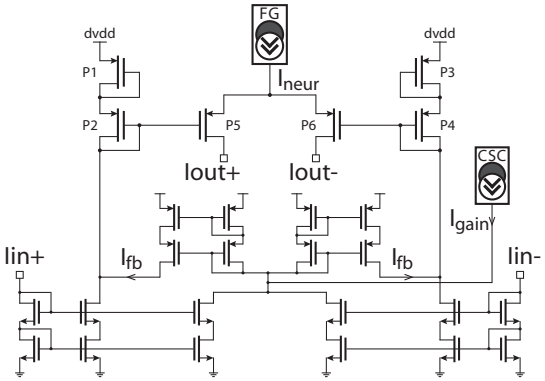


Fig. 7. Schematic of the neuron circuit. All PMOS and NMOS transistors have size 4×1 $\mu$m$^2$.

The second stage, implemented by the top part of the circuit (P1-P6), performs nonlinear transformation of the normalized input current. The common-mode signal $I_{neur}$ of the output current is set by a separate FGT.

## IV. Experimental Results

In this section, we present experimental results of the neural network chip on several classification problems. Training is performed in silicon by the chip-in-the-loop strategy using a corresponding algorithm mentioned in Section II. The learning capability is first evaluated on a standard XOR task. In the second experiment, the neural network chip is trained to distinguish faulty from functional instances of LNA circuits (parametric faults) using on-chip amplitude detector measurements. Lastly, the effectiveness of the neural classifiers to detect defects (catastrophic faults) is evaluated on a dataset of RF front-end circuits.

### A. XOR2 Problem

Learning ability evaluation and comparison with other implementations reported in literature is performed on a benchmark 2-input XOR task. It is well known that linear classifiers fail to allocate a boundary in this case. In fact, a multilayer perceptron requires a minimum of two hidden neurons for this task. For power efficiency demonstration, we limited the operating currents to 1 nA (i.e. the output currents of the GM blocks, neurons and the maximum weight currents). For the ONN classifier, the training started with just an output layer and successively added hidden layers (neurons) until all 4 patterns were classified correctly. The training consistently

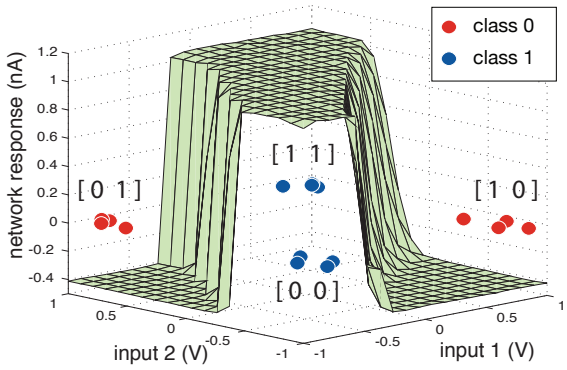


Fig. 8. Decision surface for XOR2 task obtained by measuring network output on a fine grid in input space. Also shown are the input patterns with added noise.

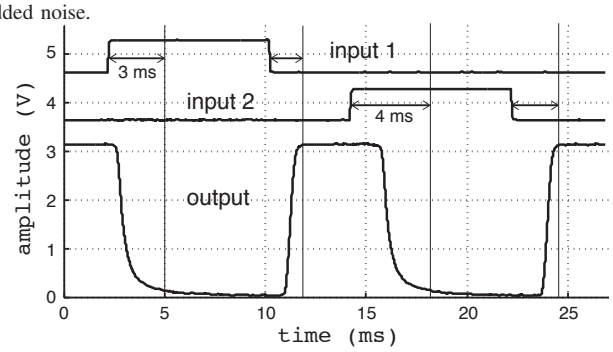


Fig. 9. Transient characteristic of the ONN trained to classify XOR2 patterns. The output is recorded from the voltage output pin of the ITOV converter.

converged with a single hidden neuron. Fig. 8 illustrates the output produced by the trained ONN with one hidden neuron. The output neuron is programmed for high gain, which explains the rail-to-rail response. The transient characteristic of the neural network is presented in Fig. 9. The response time is 4 ms, which also includes the propagation delay due to the GM and ITOV converters. The system performance and comparison data to other implementations found in the literature are summarized in Table I.

### *B. Case Study I: LNA Circuits*

In the first case study, the analog neural network is trained to separate faulty from functional LNA chips using on-chip sensor measurements in response to on-chip stimulus generators. The design and the measurement data were provided to us by the authors of [11]. The design consists of an LNA circuit integrated with two RF amplitude detectors (AD) placed at its input and output ports, as well as a voltage-controlled oscillator which provides input stimuli for BIST. An original set of 1000 devices with process variation is generated via post-layout Monte Carlo (MC) simulation (Fig. 10). On each device, we collect both a standard set of performance parameters (i.e. noise figure, gain, S11, S22, IIP3) and four AD measurements (in response to two single-tone stimuli of different power levels). The latter are used as inputs to the classifier.

The actual training and test sets are obtained via the

TABLE I

SYSTEM PERFORMANCE AND COMPARISON

| | ETANN [9] | [10] | this work |
|---|---|---|---|
| Technology | 1 um CMOS | 0.35 um, DP | 0.35 um, DP |
| Weight storage | floating gate | floating gate | FG + dynamic |
| Learning models | MLP | VMM+WTA | MLP+ONN |
| Learning strategy | off-chip | off-chip | chip-in-the-loop |
| Synapse current | 20 um @5V | 10 nA @2.4V | 2 nA @3.3V |
| Response time | 5 us | NA | 4 ms |
| Total power (XOR2 taks) | NA | 700 nW | 66 nW |
| Computation efficiency | 1.3 GMAC/s/W | 11–14 TMAC/s/W (theoretical) | 57 GMAC/s/W (measured) |

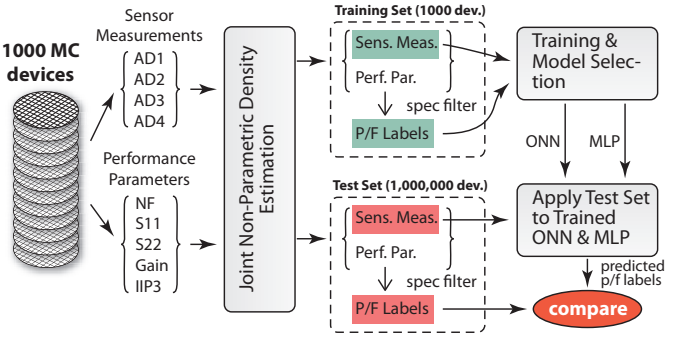


Fig. 10. Case study I. The synthetic training set is used for training and model selection. The test set is used to report classification performance.

technique described by the authors of [12]. In essence, the original dataset is used to estimate a joint non-parametric density of both the sensor measurements and the performances. This learned density is then resampled to obtain new synthetic devices which follow the original distribution. The new devices are arranged into a training set (1,000 devices) and a test set (1,000,000 devices). A large size of the test set allows us to assess the classification rate with parts per million accuracy. The performance parameters portion of the data is passed through a specification filter (defined as $Mean \pm 3 \cdot StdDev$) to obtain pass/fail labels used for training/testing. It should be noted that the training set is balanced by artificially enhancing it with faulty devices to bring the pass/fail ratio close to 1.

Besides fitting the weights, each of the presented classifiers has a model complexity parameter – the number of hidden neurons – that needs to be selected. We employ a popular technique called cross validation (CV), whereby a small portion of the original training set is reserved as validation data to test a model trained on the remaining data. This step is repeated multiple times for each number of hidden neurons with individual CV errors aggregated into a graph such as the one shown in Fig. 11. Also shown are the mean and the standard error bars of CV errors. The fact that the CV error does not improve after a few hidden neurons suggests that the optimal boundary is fairly simple. We select the best model according to the one-standard-error rule as the most parsimonious model whose score is within one standard error of the best score. For the MLP and ONN classifiers these models contain 3 and 2 hidden neurons, respectively. The resulting classifiers are retrained on the entire training set and

TABLE II

TRAINING RESULTS OF LNA CIRCUITS

| | Analog ONN | Analog MLP | Software MLP | Software MLP |
|---|---|---|---|---|
| Model size (HN) | 2 | 3 | 2 | 4 |
| Test error (PPM) | 480 | 580 | 730 | 435 |
| Die area ($mm^2$) | 0.126 | 0.138 | N/A | N/A |
| Power ($\mu$W) | 1.58 | 1.63 | N/A | N/A |

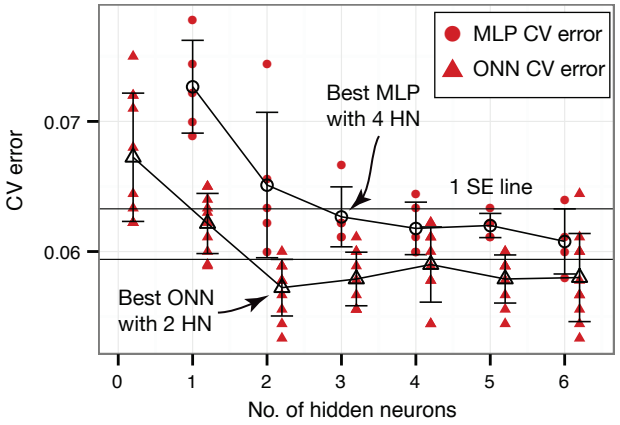


Fig. 11. Case study I: CV error vs. the number of hidden neurons for training an MLP classifier.

evaluated on the test set with the results shown in Table II. For comparison purposes, the table also reports the test error as obtained by training a software MLP with 2 and 4 hidden neurons. Although the classification accuracy is similar across various implementations, the software MLP with 4 hidden neurons and the analog ONN achieve the lowest test error. The average power consumption of the analog ONN with 2 hidden neurons is 2.6 $\mu$W. Finally, we copied the learned weights onto the FGTs and verified that the test error remained consistent within several days. A long term study of non-volatile weight retention is currently being conducted.

### C. Case Study II: Defect Filter

Identifying manufacturing defects (or catastrophic faults during the lifetime of ICs) represents a different class of problems insofar as machine learning-based test is concerned. A boundary trained to separate functional from parametric faults does not perform well on defects which do not follow the distribution of devices with process variation and appear as outliers in the space of sensor measurements. Moreover, while it is straightforward to obtain devices as affected by process variation by performing MC simulation, there are no widespread models for defect generation. Thus, relying on devices affected by process variation only, the problem of training a defect filter can be solved by constructing a boundary around the class of instances with process variation (considered as functional) serving to protect them against any defects, which appear as outliers.

For the purpose of this case study we used the data from an RF front-end chip which was provided to us by the authors of [2]. The front-end consists of an LNA, a mixer and a number

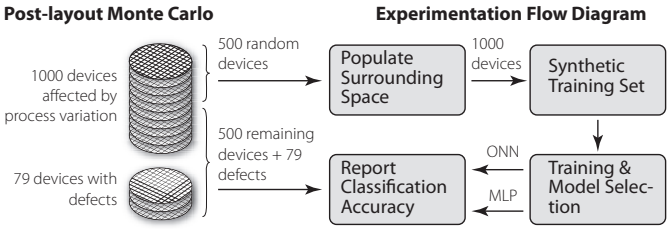


Fig. 12. Defect filter course of experiments. The synthetic training set is used for training and model selection. The test set is used to report classification performance.

of on-chip sensors serving as input features for the classification task. In particular, we consider the readings of three sensors: two envelope detectors and a DC probe. These sensors produce voltage outputs and can be conveniently interfaced with the neural classifier's inputs. The sensor measurements are obtained for 1000 devices generated through post-layout MC simulation and 79 devices with defects. The list of defects includes all possible open- and short-circuits injected one at a time at the layout level.

The course of experiments is shown in Fig. 12. The original dataset is split into a training set consisting of 500 randomly selected devices generated with process variation and a test set comprising the remaining 500 devices and all defects. Since training a neural classifier requires two classes, we generate a second class by uniformly populating the space around the 500 devices from the training set with artificial data. The objective is to leave a tiny gap between the classes for the separation boundary. Next, both analog classifiers are trained on the synthetic training set using the cross validation technique. Figures 13 and 14 illustrate training results in terms of CV errors vs. the model complexity for the MLP and ONN classifiers. The CV errors beyond 10 neurons remain at the same level and are not shown. It should be pointed out that absolute values of the CV errors mean little due to the artificial nature of the training set and are used only for model selection.

The one-standard-error line indicates that the best models for the ONN and MLP classifiers contain 4 and 8 neurons, respectively. These models are retrained on the entire synthetic set and applied to the test set with the results shown in Table III. Note that the MLP classifier's error of 1.2% is considerably lower which, however, is achieved at a larger model size. Also note that yield loss accounts for almost the entire test error for both classifiers with only a single defect being misclassified as a functional device by MLP (0.17% test escape). Fig. 15 illustrates 579 devices from the test set and a decision surface produced by the trained MLP classifier. The devices affected by process variation (blue) are located inside the cavity formed by the decision surface while the defective devices (red) are located outside. Note that only those few devices with process variation lying in the tail of distribution fall outside the enclosing boundary and, as a result, are misclassified.

TABLE III

TRAINING RESULTS ON DEFECT FILTER

| | Analog ONN | Analog MLP |
|---|---|---|
| Test error (%) | 3.79 | 1.2 |
| Test escapes (%) | 0 | 0.17 |
| Yield loss (%) | 3.79 | 1.03 |
| Model size (HN) | 4 | 8 |
| Die area ($mm^2$) | 0.21 | 0.282 |
| Power ($\mu$W) | 2.64 | 3.45 |

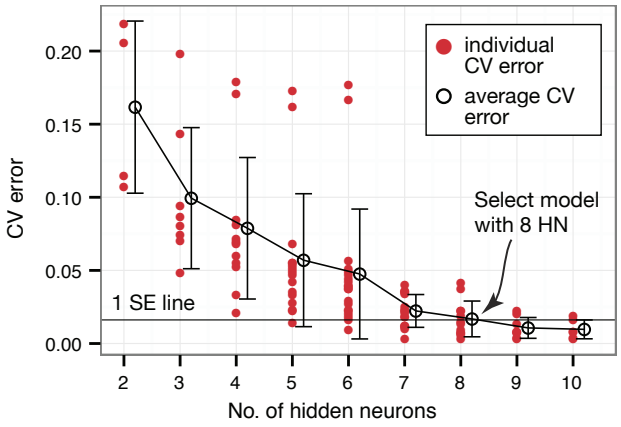


Fig. 13. Cross validation error vs. the number of hidden neurons for training a MLP classifier.

## V. CONCLUSIONS

We presented a reconfigurable neural platform supporting two classifier models programmable over various model complexities and biased over a wide range of operating currents. The circuits underlying the architecture are optimized for low cost in terms of area and power, programming flexibility and permanent non-volatile storage of learned functionality. The two classifier models demonstrated great learning ability in distinguishing faulty from functional devices due to parameter variation and identifying catastrophic faults on two realistic case studies. The presented results confirm our belief that custom analog neural classifiers can support successful decision making towards a stand-alone RF BIST solution.

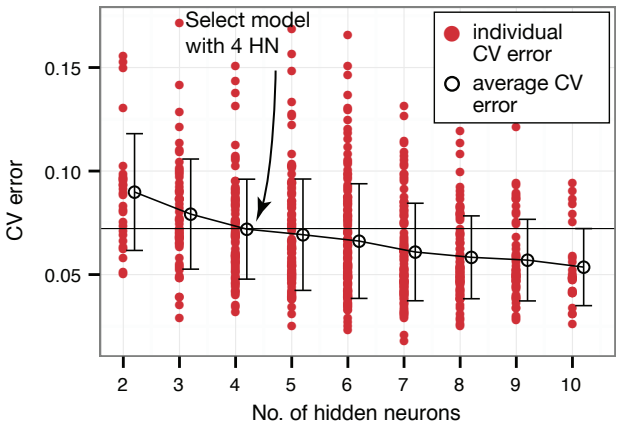


Fig. 14. Cross validation error vs. the number of hidden neurons for training an ONN classifier.

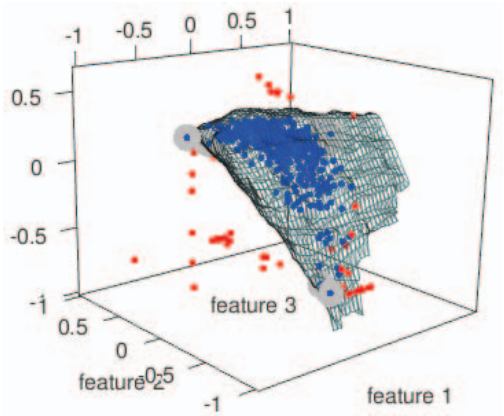


Fig. 15. Decision surface of the MLP classifier as obtained by measuring the response on a fine 3d grid of inputs and interpolating a surface over points where the response crosses a zero threshold. Misclassified devices are highlighted in gray.

## REFERENCES


[1] H.-G. 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.

[2] L. Abdallah, H.-G. Stratigopoulos, S. Mir, and C. Kelma, "RF front-end test using built-in sensors," *IEEE Design & Test of Computers*, vol. 28, no. 6, pp. 76–84, 2011.

[3] C. Schlottmann and P. Hasler, "A highly dense, low power, programmable analog vector-matrix multiplier: The FPAA implementation," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 1, no. 3, pp. 403–411, 2011.

[4] A. Bandyopadhyay, G. J. Serrano, and P. Hasler, "Adaptive algorithm using hot-electron injection for programming analog computational memory elements within 0.2% of accuracy over 3.5 decades," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 9, pp. 2107–2114, 2006.

[5] D. Maliuk, H.-G. Stratigopoulos, H. He, and Y. Makris, "Analog neural network design for RF built-in self-test," in *Proceedings of the IEEE International Test Conference (ITC)*, 2010, pp. 23.2.1–23.2.10.

[6] D. Maliuk, N. Kupp, and Y. Makris, "Towards a fully stand-alone analog/RF BIST: A cost-effective implementation of a neural classifier," *IEEE VLSI Test Symposium*, pp. 62–67, 2012.

[7] S. E. Fahlman and C. Lebiere, "The cascade-correlation learning architecture," in *Proc. Advances Neural Inform. Process. Syst*, 1990, vol. 2, pp. 524–532.

[8] M. Riedmiller, "Advanced supervised learning in multi-layer perceptrons from backpropagation to adaptive learning algorithms," *Computer Standards and Interfaces*, vol. 16, no. 3, pp. 265–278, 1994.

[9] H. A. Castro, S. M. Tam, and M. A. Holler, "Implementation and performance of an analog nonvolatile neural network," *Analog Integrated Circuits and Signal Processing*, vol. 4, no. 2, pp. 97–113, 1993.

[10] S. Ramakrishnan and P. Hasler, "Vector-matrix multiply and winner-take-all as an analog classifier," *IEEE Transactions on Very Large Scale Integration Systems*, (available online), 2013.

[11] N. Kupp, H. Huang, P. Drineas, and Y. Makris, "Improving analog and RF device yield through performance calibration," *IEEE Design & Test of Computers*, vol. 28, no. 3, pp. 64–75, 2011.

[12] H.-G. Stratigopoulos, S. Mir, and Y. Makris, "Enrichment of limited training sets in machine-learning-based analog/RF test," in *Design, Automation and Test in Europe*, 2009, pp. 75–108.