On-Chip Intelligence: A Pathway to Self-Testable, Tunable, and Trusted Analog/RF ICs

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*— This paper discusses the design of an experimentation platform intended for prototyping low-cost neural networks for on-chip integration, towards supporting built-in self-test, post-production self-calibration, and trust evaluation capabilities. Particular emphasis is given to cost-efficient implementation reflected in stringent area and power constraints of circuits dedicated to neural networks, which, however, should not compromise their learning ability and correct functionality throughout their lifecycle. Our chip consists of a reconfigurable array of synapses and neurons operating below threshold and featuring sub-μW power consumption. The synapse circuits employ dual-mode weight storage: (1) a dynamic mode, for fast bidirectional weight updates during training and (2) a non-volatile mode, for permanent storage of learned functionality. The chip architecture supports two learning models: a multilayer perceptron and an ontogenic neural network. The system performance and learning ability are evaluated on the XOR2 benchmark.**

I. Introduction

On-chip trainable neural networks hold great promise in enabling various desired features of modern integrated circuits (IC). For example, circuit deployment in safety-critical applications calls for a built-in self-test (BIST) method [1] such that the chip can assess its own functional health and issue alerts in case of malfunctions. Similarly, contemporary analog/RF ICs take an aggressive design approach in order to maximize performance, possibly at the expense of robustness, which is later compensated for through on-chip self-healing hardware [2]. As another example, security concerns regarding the globalized IC supply chain, which may be vulnerable to malicious attacks (a.k.a. Hardware Trojans), have sparked interest in adding hardware for monitoring operation trustworthiness [3].

In the heart of these problems lies some form of low-cost on-chip intelligence, which acquires measurements through low-cost sensors [4] and makes pass/fail decisions regarding correctness (see Fig. 1) or trustworthiness (see Fig. 2), or selects appropriate tuning knob positions to ensure specification-compliant functionality (see Fig. 3). Neural networks have proven to be adequate learning models possessing the power to extract this kind of information from low-cost measurements [5]. Prior research has explored their software implementation running on an external computer [6] or a built-in DSP block [7]. However, such processing resources are not always available to a stand-alone IC, calling for a low-cost hardware neural network which can be integrated on-chip in order to provide the aforementioned capabilities.

The nature of the described applications imposes rather strict requirements on a hardware implementation: small area and low power of circuits dedicated to neural networks, rapidly programmable weight memory for fast training, and long-term non-volatile storage of learned weights throughout the network's lifetime. Accordingly, we selected an analog circuit implementation for its superior power efficiency during run time, compact size and the possibility of permanent weight storage using analog floating gate (FG) memory. Indeed, when high precision is not required, analog computation can be as much as 1000x more energy efficient than digital [8]. 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 weight storage with high accuracy [9].

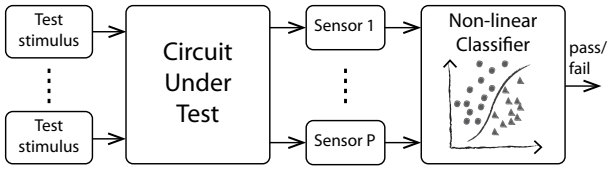

Fig. 1. Components of the BIST architecture. The circuit under test (CUT) is excited by test stimuli. Multiple sensors collect simple measurements which are processed by an on-chip non-linear classifier that produces a pass/fail test result.

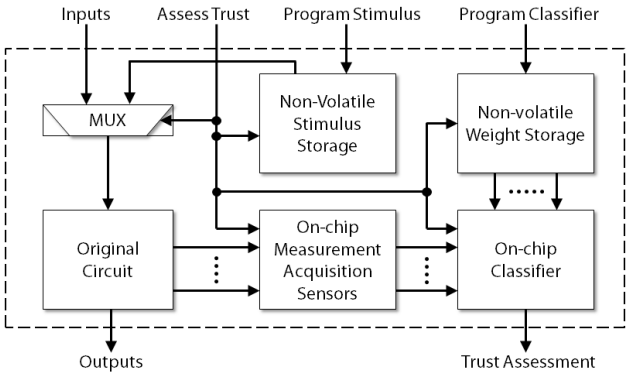

Fig. 2. Components of the self trust-evaluation. The CUT is excited by test stimuli. Multiple sensors collect simple measurements which are processed by an on-chip non-linear classifier to decide whether a hardware Trojan exists.

Following the mentioned guidelines we have designed and fabricated a reconfigurable experimentation platform in a 0.35-μm CMOS process available from TSMC. The remainder of the paper is structured as follows. Section II introduces the ONN learning model. Section III describes the implementation details including the overall architecture, the weight storage mechanism, the synapse and neuron circuits. Section IV presents the measurement data and evaluates the system's learning ability on the typical XOR-2 classification task.

978-1-4799-4132-2/14/$31.00 ©2014 IEEE

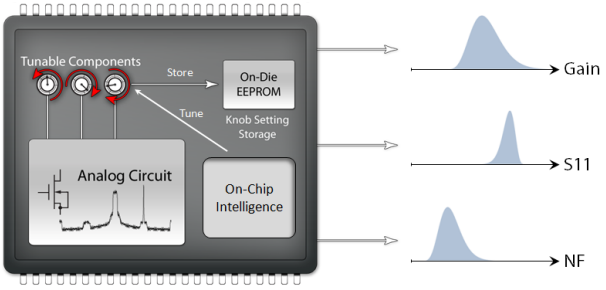

Fig. 3. Components of a self-tunable chip. Knobs are tuned by an on-chip intelligent entity, such as a neural network, as a result of observing the circuit response to a given test stimulus via on-chip sensors. Chosen knob settings are stored on-chip using non-volatile memory.

II. Overview of Ontogenic Neural Networks

Fig. 4 illustrates a block diagram of the ONN learning model. A decision boundary is constructed by successively adding hidden neurons; 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 [10], which repeats the following steps for each added neuron. Suppose that our current stage has $M-1$ hidden neurons, as shown in Fig. 4. 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 or of the output neuron undergo modification, 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 a classification error on a validation set. The correlation maximization and the error minimization are done by the resilient back propagation algorithm (iRPROP+), which can be efficiently customized for hardware networks.

III. Chip Design

A. System Description

Fig. 5 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. The reconfigurable fabric for the ONN topologies is a subset of synapses lying below the main diagonal (similar to the layout in Fig. 4). Global connectivity is programmable by means of multiplexors inserted between rows. The core operates in the analog domain

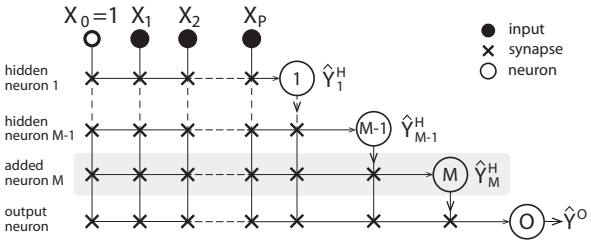

Fig. 4. The ontogenic neural network topology. The hidden neurons receive connections from both the network inputs X_i and the outputs of the previous hidden neurons $\widehat{Y}_j^H$. The bottom neuron serves as a network output $\widehat{Y}^O$.

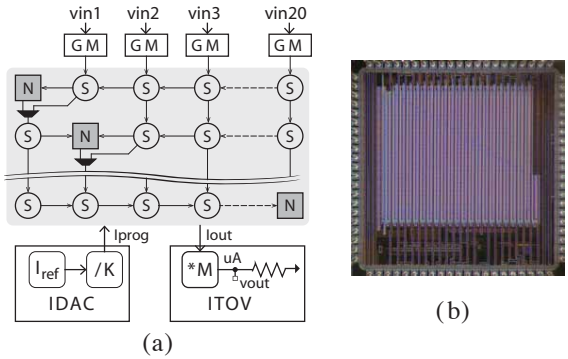

Fig. 5. (a) System architecture of the ontogenic neural network chip. The reconfigurable array of synapses (S) and neurons (N) is shown in the shaded box. (b) Die photograph of ONN chip measuring 3×3 mm^2.

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: the dynamic, for fast bidirectional weight updates, and the non-volatile, for long-term storage of learned weights. The dynamic mode is engaged during training, when the weight values undergo multiple updates. Upon the 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. 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 the reading/sourcing accuracy of at least 8 bits.

B. Weight Storage

The principle of weight storage is illustrated in Fig. 6. 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

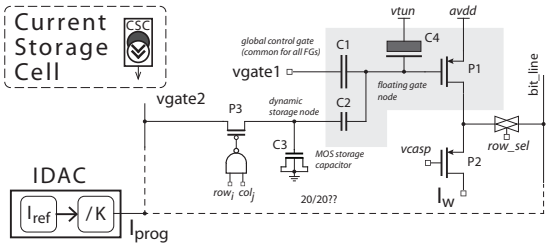

Fig. 6. Schematic of the current storage cell along with the dynamic programming loop. The sizes of key components are as follows: P1 = 2×2 μm^2, P2 = 2×1 μm^2, P3 = 0.4×0.35 μm^2; C1 = 40 fF, C2 = 15 fF, C3 = 1.35 pF, C4 = 0.6 fF.

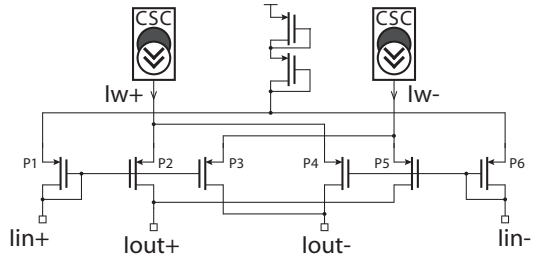

Fig. 7. Schematic of the synapse circuit (P1 = ... = P6 = 4×2 μm^2).

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*. The details of non-volatile and dynamic programming are described in [11].

C. Synapse and Neuron Circuits

The synapse circuit, illustrated in Fig. 7, 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 components storage and a six-transistor core P1-P6. The neuron circuit, illustrated in Fig. 8, applies a nonlinear activation function to the sum of the outputs of the connected synapses. The 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. The top part (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

A. Weight Decay

Errors in dynamic programming directly affect the learning ability of the analog neural network. Reverse-bias leakage current of the switch transistor P3 (Fig. 6) is particularly important since it determines the time window during which the weight change remains insignificant. In order to characterize this effect we measured weight decays for a randomly selected set of CSCs. The weight leakage rate r_w can be defined as a relative change of the weight value per unit time, or

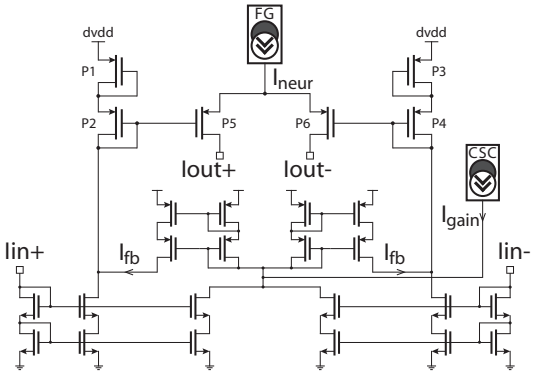

Fig. 8. Schematic of the neuron circuit. All PMOS and NMOS transistors have size 4×1 μm^2.

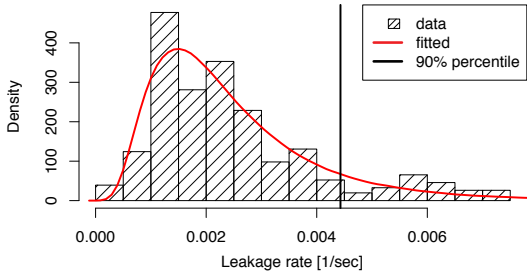

Fig. 9. A histogram of weight leakage rate estimates with a corresponding fitted lognormal density function.

$$r_w = \Delta I_w / (\Delta t \cdot I_w) \quad (1)$$

where I_w is the original weight current, ΔI_w is the weight change over time Δt. Fig. 9 shows a histogram of the leakage rate estimates collected for 35 randomly selected CSCs. The drain current of each CSC was dynamically programmed to a number of values from 0.1 nA to 30 nA. The normal distribution of the process variation parameters suggests that the leakage rate belongs to the family of lognormal distributions (leakage current affects the exponential part of the output drain current). Indeed, Kolmogorov-Smirnov test does not reject the null hypothesis (p-value=0.8). The leakage rate corresponding to the 90% percentile is 0.00443 sec^{-1}. If we assume that the weight value is represented by an 8-bit word, a 1-bit change occurs in 0.88 sec. It has been observed that the change of 1-bit results in virtually no change in the network output. To limit the exposure to weight decay, large datasets (>200 observations) are split into smaller chunks during the forward pass and separated by reprogramming the dynamic memory.

B. XOR2 Problem

We begin with the classical 2-input XOR task. It is well known that linear classifiers fail to allocate a boundary in this case. In fact, a multilayer perceptron (MLP) 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). The training started with just an output layer and successively added hidden layers (neurons) until all 4 patterns were classified correctly. The

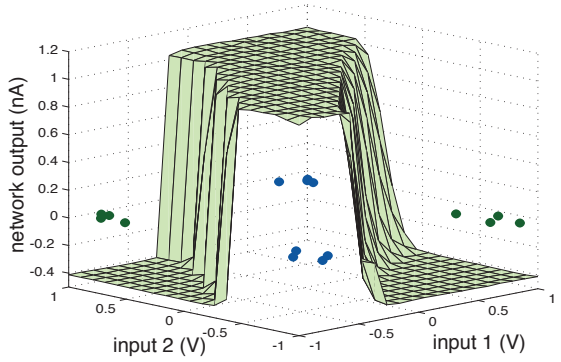

Fig. 10. Decision surface for XOR2 task obtained by measuring network output on a fine grid in input space. Also shown are the input patterns with added noise.

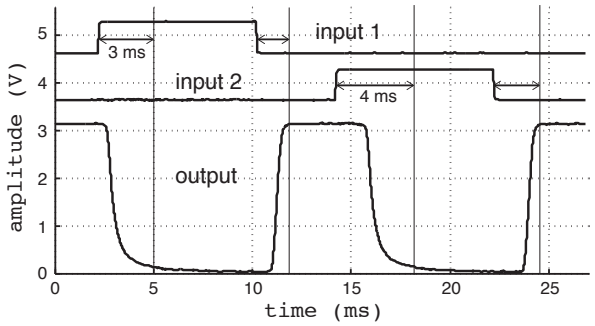

Fig. 11. Measured transient characteristic of the ONN trained to classify XOR2 patterns. The output is recorded from the voltage output pin of the ITOV converter.

training consistently converged with a single hidden neuron. Fig. 10 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. 11. 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 are summarized in Table I.

V. Conclusions

The system characteristics of the presented analog implementation of the ONN make it a great candidate for on-chip inclusion as a part of a BIST, self-calibration, or self trust evaluation process. Low power is achieved by biasing all circuits below threshold and employing the translinear principle for analog computation. Efficient dynamic weight programming is achieved by storing dynamic voltage on a MOS capacitor which drives a low-coupling gate of a mutliple-input FGT. Non-volatile weight storage is made possible through the FG technology available in double-poly CMOS. Finally, the ONN chip demonstrated excellent results in solving the benchmark XOR-2 task. Future work will focus on designing and fabricating custom analog/RF ICs that incorporate small, appropriately configured portions of our platform, in order to demonstrate the efficiency of analog neural networks in providing solutions to the aforementioned problems.

TABLE I

System performance and comparison

	ETANN [12]	[13]	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)

References

[1] 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.

[2] N. Kupp, H. Huang, P. Drineas, and Y. Makris, "Post-production performance calibration in analog/RF devices," in *Proceedings of the IEEE International Test Conference (ITC)*, 2010, pp. 8.3.1–8.3.10.

[3] Y. Jin, D. Maliuk, and Y. Makris, "Post-deployment trust evaluation in wireless cryptographic ICs," in *IEEE Design & Test of Computers (DATE)*, 2012, pp. 965–970.

[4] 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.

[5] H.-G. Stratigopoulos and Y. Makris, "Nonlinear 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.

[6] 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.

[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] 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.

[9] 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.

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

[11] D. Maliuk and Y. Makris, "A dual-mode weight storage analog neural network platform for on-chip applications," in *IEEE International Symposium on Circuits and Systems*, 2012, pp. 2889–2892.

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

[13] S. Ramakrishnan and P. Hasler, "Vector-matrix multiply and winner-take-all as an analog classifier," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 22, no. 2, pp. 353–361, 2014.