A Dual-Mode Weight Storage Analog Neural Network Platform for On-Chip Applications

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*— On-chip trainable neural networks show great promise in enabling various desired features of modern integrated circuits (IC), such as Built-In Self-Test (BIST), security and trust monitoring, self-healing, etc. Cost-efficient implementation of these features imposes strict area and power constraints on the circuits dedicated to neural networks, which, however, should not compromise their ability to learn fast and retain functionality throughout their lifecycle. To this end, we have designed and fabricated a reconfigurable analog neural network (ANN) chip which serves as an expertise acquisition platform for various applications requiring on-chip ANN integration. With this platform, we intend to address the key cost-efficiency issues: a fully analog implementation with strict area and power budgets, a learning ability of the proposed architecture, fast dynamic programming of the weight memory during training, and high precision non-volatile storage of weight coefficients during operation or standby. We explore two learning structures: a multilayer perceptron (MLP) and an ontogenic neural network with their corresponding training algorithms. The core circuits are biased in weak inversion and make use of the translinear principle for multiplication and non-linear conversion operations. The chip is mounted on a custom PCB and connected to a computer for chip-in-the-loop training. We present measured results of the core circuits and the dual-mode weight memory. The learning ability is evaluated on a 3-input XOR classification task.**

I. Introduction

Various realities of modern semiconductor manufacturing business necessitate the inclusion of dedicated on-chip circuitry for post-deployment monitoring – and potentially action-taking – of analog/RF ICs, in order to enhance their reliability, robustness, and trustworthiness. For example, circuit deployment in mission-critical applications (e.g. avionics, medicine) and sensitive environments (e.g. space) calls for a 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-calibration and self-healing hardware [2] (i.e. tuning knobs). Finally, security concerns regarding the globalized IC supply chain, which may be vulnerable to malicious attacks (a.k.a. Hardware Trojans [3]), have sparked interest in adding hardware for monitoring operation trustworthiness. In the heart of these three problems, lies some form of low-cost on-chip intelligence, which acquires measurements through low-cost sensors and makes pass/fail decisions regarding correctness/trustworthiness, or selects appropriate tuning knob positions to ensure specification-compliant functionality. To this end, our research focuses on developing a low-cost analog neural network which can be integrated with the circuit in order to provide the aforementioned reliability, robustness and trustworthiness capabilities.

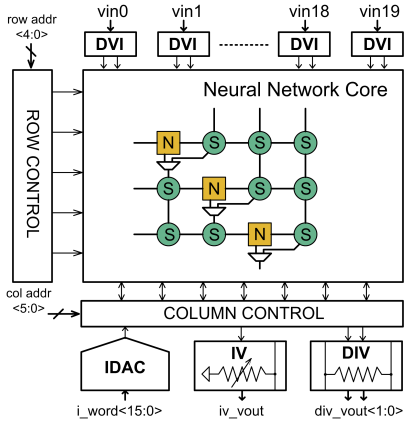

Fig. 1. Top level chip architecture. The core includes synapses (S), neurons (N) and multiplexors for topology configuration. Peripheral circuits support network operation and programming.

The nature of the described applications imposes rather strict requirements on the choice of neural network implementation. More specifically, the circuits dedicated to the ANN should incur low area and power overhead, compared to the system they are built into. The weight memory should be rapidly programmable during training and also feature long-term storage of the learned weights. The ANN should have a high degree of learning capacity, which is determined by the weight and signal dynamic range, weight programming accuracy, synapse linearity, noise level, etc. Finally, the implementation technology must be the same as for the other circuits for seamless integration into a single chip. Note that the outlined requirements exclude digital implementation due to its large area overhead.

In the past, we proposed a mixed-signal implementation of a neural network and successfully demonstrated its learning ability on real-life classification problems [1], [4]. Local SRAM cells were used to store weight coefficients in digital form and multiplying digital-to-analog converter-based synapses were used to convert them into analog values with further multiplication. Off-chip memory was required to permanently store the weights and copy them into the chip's memory on power-up. This work represents a major improvement of our previous design, focusing on cost-effective implementation for on-chip integration. In effect, we have designed and fabricated a reconfigurable neural network platform featuring 20 inputs, a 30×20 array of synapses and 20 neurons. The chip provides us with an experimentation ground to explore various learning models and their topologies as well as strategies to train them. Performing model selection on a given learning problem allows us to find the most compact implementation achieving the desired performance. This model can later be customized and integrated on-chip for a specific application.

978-1-4673-0219-7/12/$31.00 ©2012 IEEE

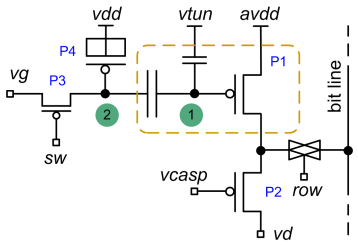

Fig. 2. Current storage cell.

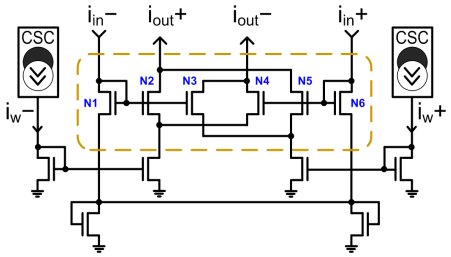

Fig. 3. Synapse circuit.

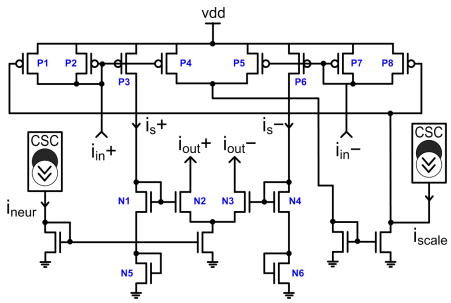

Fig. 4. Neuron circuit.

II. Design Overview

A. System Description

The block diagram of our ANN is presented in Figure 1. The core of the design is a 30×20 array of synapses (S); each row is locally connected to a corresponding neuron (N). Global connectivity is programmable by means of multiplexors inserted between rows. This allows the core to be configured into several learning structures, including a multilayer perceptron [5] and an ontogenic neural network [6]. The former is a three-layer network of fixed topology with programmable number of neurons in each layer. By contrast, the ontogenic configuration allows for the network topology to be learned dynamically in parallel to its weights.

The information processing inside the core is analog; the signals and weights are represented by balanced differential currents. The current signal domain and the translinear principle offer a wide variety of mathematical functions, including multiplication and *tanh*-like transformation [7], whereas the differential coding allows for four-quadrant multiplication. A single weight value requires two current sources for differential current storage. It appears that the overall learning ability depends – to a great extent – on the "quality" of these sources. The ideal implementation should have the following characteristics: high precision, non-volatile storage and fast bidirectional update. To this end, we designed a novel current storage cell (CSC) featuring two modes of weight storage: dynamic, for rapid bidirectional update, and non-volatile, for long-term storage of learned weights. The dynamic mode is engaged during training, when the weight values undergo multiple changes. Once the best set of weights is found, their values are copied onto the floating gate transistors for permanent storage.

Surrounding the core are the peripheral circuits providing support for fast programming, configuration storage, and interfacing with the external world. In particular, the "DVI" blocks convert voltage-encoded input signals into balanced differential currents required by the core. Not only does it simplify the interface with the off-chip stimuli generator, but it also allows a direct connection of on-chip sensors with voltage output. The row and column controls isolate individual CSC cells from the array for weight programming. Finally, the circuits at the bottom facilitate network training by transferring some of the programming related tasks on-chip. In particular, a digitally-controlled current source "IDAC" generates target currents for dynamic programming of the CSC. Both the "IV" and the "DIV" blocks convert the output current supplied by the core into voltage, which is captured by an off-chip ADC. For high accuracy we use the single-ended current to voltage converter "IV", which is necessary for floating-gate transistor (FGT) programming. This block constitutes a part of a fast current measurement system; the current values are derived from the measured voltages using characterization data of the converter. The "DIV" block converts differential currents produced by the network output into differential voltages. Although less accurate, this is useful for quick network output evaluation in *run* mode. During training, however, accurate estimation of the error between the network output and the target value is necessary, which can only be furnished by the "IV" converter.

B. Weight Storage Mechanism

The principle of our CSC, a building block of the weight memory, is presented in Figure 2. At the core of the circuit is the floating gate (FG) PMOS transistor P1 (enclosed by the dashed box), whose drain current is the entity being stored. The drain current is modulated by the voltage on the FG (*node 1*), which is itself determined by the FG node charge and the gate voltage (*node 2*). Manipulating the charge value constitutes the non-volatile storage mode. For dynamic storage of the gate voltage we use a simple sample-and-hold (S/H) circuit comprising a MOS capacitor P4 and a switch transistor P3. The cascode transistor P2 is inserted to minimize the drain coupling effect to the FG node and also to isolate the drain from the main circuit during programming.

We selected two mechanisms for non-volatile programming of FGTs. 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. Although the two mechanisms allow for bidirectional charge transfer, due to the difficulties in on-chip routing of high voltages and poor controllability we use FN tunneling for global erase. Injection, on the other hand, is used to program individual FGTs to a target current with high accuracy. First, the FGT of interest is isolated from the containing circuitry by raising the global *vcasp* and connecting its drain to the bit line. Next, we ramp up the *avdd* and apply a series of short pulses to the bit line, measuring the drain current with the "IV" circuit after each pulse. The amount of charge injected during each pulse depends on both the source-to-drain voltage and the duration of the pulse. For accurate injection we adopt the algorithm described in [8], however,

using a pulse-width instead of a drain voltage modulation.

For fast bidirectional weight updates during training we use the dynamic storage. In this mode, the CSC of interest is isolated from the array, the gate is shorted to the drain, while the drain is connected to the "IDAC" block. The "IDAC" is programmed to sink the desired current, thereby charging the dynamic capacitor P4 and forcing the diode-connected FG transistor to supply equal current. Once this self-biasing loop is stabilized the switch transistor disconnects the gate. Crucial items in this programming scheme are the accuracy of the "IDAC" current source and the switch non-idealities. While the S/H circuit is kept very simple, we perform thorough calibration of the "IDAC" block to match the accuracy of dynamic programming to that of FGT injection.

C. Synapse Circuit

The synapse circuit, illustrated in Figure 3, implements a four-quadrant multiplication function [9]. The circuit features two CSC cells for differential weight components storage and a six-transistor core, enclosed by the dashed box. Provided the core transistors are identical and there is no mismatch, the output differential and common mode currents can be obtained by

$$\begin{aligned} i_{out}^{+} - i_{out}^{-} &= \frac{i_{in}^{+} - i_{in}^{-}}{i_{in}^{+} + i_{in}^{-}}(i_{w}^{+} - i_{w}^{-}) \\ i_{out}^{+} + i_{out}^{-} &= i_{w}^{+} + i_{w}^{-} \end{aligned} \tag{1}$$

where i_{in}^{+}, i_{in}^{-} are the differential components of the input signal and i_{w}^{+}, i_{w}^{-} are the differential components of the weight value. The core results in a very compact layout, while most of the area is occupied by the CSC cells due to the dynamic capacitors (approximately 1 pF each).

D. Neuron Circuit

The neuron circuit, illustrated in Figure 4, applies a non-linear activation function to the sum of the outputs of the connected synapses. For summation, the positive and negative components of the synaptic outputs are connected to the corresponding i_{in}^{+} and i_{in}^{-} terminals of the neuron circuit. The non-linear transformation is completed in two stages. The first stage, represented by P1-P8 and the feedback loop, performs input normalization and gain control. In particular, the feedback loop adjusts the common mode to the value i_{scale}, which is stored in the CSC. This also affects the slope of the activation function. We use this property to adjust the neuron to the number of synapses, as well as to the specifics of a learning task (i.e. regression or classification).

Non-linear transformation of the normalized current $i_{s}^{+} - i_{s}^{-}$ is performed by the second stage [9]. Assuming the transistors N1-N6 are identical and utilizing the translinear principle, the output differential and common mode currents are obtained by

$$\begin{aligned} i_{out}^{+} - i_{out}^{-} &= \frac{(i_{s}^{+})^{\frac{1+\kappa}{\kappa}} - (i_{s}^{-})^{\frac{1+\kappa}{\kappa}}}{(i_{s}^{+})^{\frac{1+\kappa}{\kappa}} + (i_{s}^{-})^{\frac{1+\kappa}{\kappa}}} \cdot i_{neur} \\ i_{out}^{+} + i_{out}^{-} &= i_{neur} \end{aligned} \tag{2}$$

where κ is the subthreshold slope; i_{neur} sets the common mode current of the neuron output.

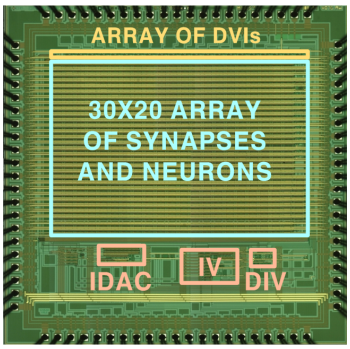

Fig. 5. Die photo of the neural network chip implemented in 0.5-μm process.

E. Learning Strategy

Training is performed in a hierarchical arrangement using chip-in-the-loop approach. At the bottom level is the hardware neural network with the low-level programming support by the peripheral circuitry. The next level is a custom PCB housing our chip and other off-the-shelf components that provide analog, digital and high voltage interfaces to the chip. A commercial FPGA board implements a communication link between a host computer and the custom PCB. Finally, the software layer (in MATLAB) implements training algorithms for different network configurations.

For the MLP structure we adopt an improved resilient back-propagation algorithm (iRPROP+) [10]. Originating from the family of first-order gradient descent algorithms, it uses weight perturbation for gradient estimation and local learning rate adaptation for faster training. When the chip is configured as an ontogenic network we employ a cascade-correlation algorithm [11]. The algorithm not only optimizes weight values, but also searches for the best architecture by progressively adding more neurons to the existing topology.

III. Experimental Results

The die photograph of the chip fabricated in a 0.5-μm CMOS technology is shown in Figure 5. The core of the chip features 20 inputs, 30 neurons and 600 synapses, occupying 1.5×2.5 mm^2 die area. Such a large number of components makes it a flexible platform to explore models of various scales. The final models to be integrated on chip for the target applications will be much smaller. The power consumption largely depends on the chip configuration and the common mode current values. As a first step, we calibrated and tested all of our peripheral components using external instruments. The achieved linearity for the "IDAC" and "IV" blocks for currents in the range from 0.5 to 200 nA was around **8 bits**.

Next, we evaluated the efficiency of our CSC memory. For the non-volatile storage, we characterized a random set of FG transistors to build an injection model. Using this model we could predict the programming pulse length (ranging from 1 μs to 300 ms) producing the desired increase in a drain current. For our range of interest (from 0.5 to 200 nA) the estimated programming error was below 0.5 nA resulting in the **accuracy of more than 8 bits**. For the dynamic storage, we evaluated the leakage rate, caused by the reverse biased junction of the switch and resulting in weight distortion over time. It appeared that the leakage currents depend on the stored

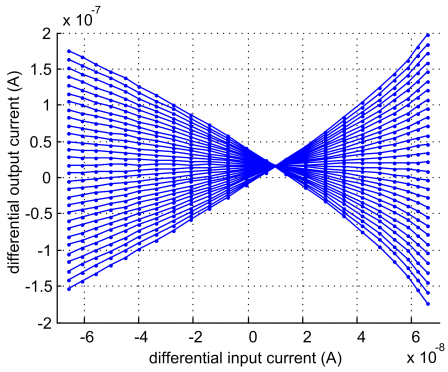

Fig. 6. Synapse multiplication characteristic. The weight values are stored in dynamic mode covering the range from -200 to 200 nA.

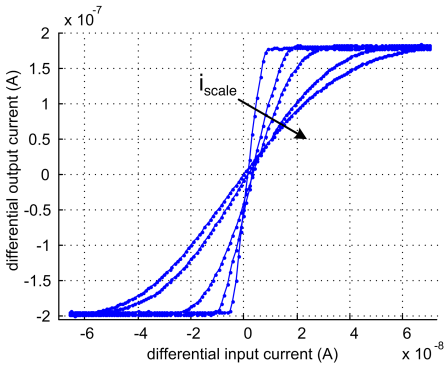

Fig. 7. Neuron transfer characteristic. i_{scale} is from (5, 25, 50, 100, 200) nA. Larger i_{scale} results in flatter characteristic.

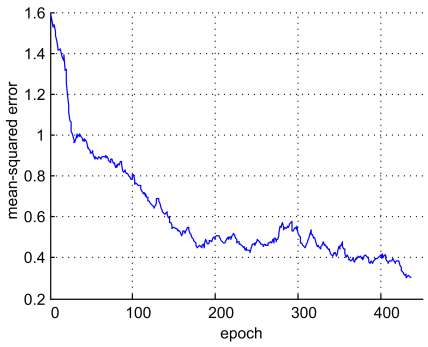

Fig. 8. XOR3 learning curve of the hardware neural network configured as a MLP with 3 input, 3 hidden and 1 output layers.

voltage with the worst case voltage change rate of 1.6 mV/s. In terms of the CSC's output, 1 bit current change occurs in 200 ms for the worst case output current of 200 nA. Periodic refresh of the dynamic memory is performed when the training algorithm updates the weights or between long forward passes.

Figure 6 shows the measured multiplication characteristic of the synapse circuit for 30 equidistant weight values from -200 to 200 nA. The input differential current is produced by the "DVI" converter, while the input and output currents are measured by the calibrated "IV" block. The observed non-linearity and DC offset can be attributed to transistor mismatch and process variations. For the purpose of training, however, this is not a major limitation, because our learning strategy does not assume a linear model for gradient descent. Figure 7 illustrates the measured characteristic of the neuron circuit for i_{neur} set to 200 nA and different values of i_{scale}. As expected, for low values of i_{scale} the characteristic resembles a step-like function, becoming more flat for high values. We use this property in training: the gain is set low in the beginning for better sensitivity of gradient measurements and is gradually increased towards the end of training. The gain of the output neuron is set to maximum at this point for binary decisions.

Finally, we present the results of learning for a popular 3-input XOR problem. The training set consists of 8 vectors covering all possible input combinations. We do not divide it into separate training and validations sets; our task is to demonstrate the chip's ability to learn complex boundaries, which can not be achieved with linear classifiers. The learning structure we selected is a MLP with 3 inputs, 3 hidden neurons, and one output neuron. Initialized with random weight values, the training proceeds by descending down the error surface, as illustrated in Figure 8. The algorithm successfully escapes a region of local minimum between 200 and 300 epochs and converges after 437 epochs, achieving 100% classification rate on the training sample.

IV. Conclusions

We introduced the design of a reconfigurable ANN platform featuring a novel dual-mode weight storage memory. The system was fabricated in a 0.5-μm CMOS process and mounted onto a custom PCB. A hierarchical chip-in-the-loop setup was used to perform learning. We presented data from the measured synapse and neuron characteristics, dynamic and non-volatile weight programming, and demonstrated its learning ability on a XOR3 problem. Future work includes applying the platform to real-life classification problems for analog/RF BIST and also fine tuning the CSC circuit for better performance.

V. Acknowledgements

The authors would like to thank Dr. Paul Hasler and his group at Georgia Institute of Technology for their assistance with the FG transistors technology.

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 and Y. Makris, "Hardware trojans in wireless cryptographic integrated circuits," in *Special issue of IEEE Design & Test of Computers (D&T)*, 2010, vol. 27, pp. 26–35.

[4] D. Maliuk, H.-G. Stratigopoulos, and Y. Makris, "An analog VLSI multilayer perceptron and its application towards built-in self-test in analog circuits," in *IEEE International On-Line Test Synposium (IOLTS)*, 2010, pp. 71–76.

[5] S. Haykin, *Neural Networks: A Comprehensive Foundation (2nd Edition)*, Prentice Hall, 1998.

[6] V. Honavar and L. Uhr, "Generative learning structures and processes for generalized connectionist networks," in *Inf. Sci.*, 1993, vol. 70.

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

[8] 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.

[9] M. Valle and F. Diotalevi, "A dedicated very low power analog VLSI architecture for smart adaptive systems," in *Applied Soft Computing 4*, 2004, pp. 206–226.

[10] R. Smithies, S. Salhi, and N. Queen, "Adaptive hybrid learning for neural networks," *Neural Computation*, vol. 16, no. 1, pp. 139–157, 2004.

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