An Analogue Neuromorphic Co-Processor That Utilizes Device Mismatch for Learning Applications

Chetan Singh Thakur, *Member, IEEE*, Runchun Wang, *Member, IEEE*, Tara Julia Hamilton, *Member, IEEE*, Ralph Etienne-Cummings, *Fellow, IEEE*, Jonathan Tapson, *Member, IEEE*, and André van Schaik, *Fellow, IEEE*

***Abstract*— As the integrated circuit (IC) technology advances into smaller nanometre feature sizes, a fixed-error noise known as device mismatch is introduced owing to the dissimilarity between transistors, and this degrades the accuracy of analog circuits. In this paper, we present an analog co-processor that uses this fixed-pattern noise to its advantage to perform complex computation. This circuit is an extension of our previously published trainable analogue block (TAB) framework and uses multiple inputs that substantially increase functionality. We present measurement results of our two-input analogue co-processor built using a 130-nm process technology and show its learning capabilities for regression and classification tasks. We also show that the co-processor, comprised of 100 neurons, is a low-power system with a power dissipation of only 1.1 μW. The IC fabrication process contributes to randomness and variability in ICs, and we show that random device mismatch is favorable for the learning capability of our system as it causes variability among the neuronal tuning curves. The low-power capability of our framework makes it suitable for use in various battery-powered applications ranging from biomedical to military as a front-end analog co-processor.**

***Index Terms*— Neuromorphic engineering, analogue integrated circuit design, stochastic electronics, neural network hardware.**

I. Introduction

THE shrinking transistor feature sizes have enabled an increase in switching speeds and memory density, leading to rapid improvements in computer performance over the last few decades [1]. At the same time, as technology improvements push us closer to the physical limits of semiconductor devices, all kinds of process randomness tend to have larger effects on their performance. This randomness includes mismatch in device dimensions and doping concentrations that occur at the nanoscale owing to limitations of the fabrication process.

Biological systems have been able to overcome many of the problems similar to those being faced by IC designers, to deliver reliable, real-time computation in neural circuits. Although built from low-performance components, these neural circuits have themselves been pushed to the extreme physical limits of their 'feature size' by evolution. This serves as a motivation for the investigation of alternative electronic and computational architectures based on neurobiological systems [2]. The goal of neuromorphic engineering is to build systems that match the performance of biological systems in challenging tasks such as vision [3], [4].

Our co-processor is inspired from the population coding present in the nervous system [5]. In this co-processor, which is based on the Trainable Analogue Block (TAB) framework, physical quantities are encoded into a population of neurons using their tuning curves. In our system, the inputs are voltage signals, which could be outputs from an array of sensors representing physical quantities in the physical world. In a biological system, neurons within the same cortical column have highly heterogeneous responses to the same input stimulus. The heterogeneity of neuronal responses has been thought to be beneficial for sensory coding when stimuli are decoded from the population response [6], [7]. The shape of tuning curves of individual neurons and the heterogeneity of neuronal responses both affect the quality of population coding and the accuracy of information processing in the cortex [8]. We have adopted a similar concept in our co-processor by using a heterogeneous population of neurons.

Device mismatch has previously been utilised for computation, as proposed by [9]–[12]. Several researchers have proposed architectures similar to the TAB framework, based on the Neural Engineering Framework (NEF) [13] or the Extreme Learning Machine (ELM) [14]. These frameworks were implemented using spiking neurons that processed the spike inputs [15]–[17]. Cameron *et al*. [9] used spike timing-dependent plasticity to correct process mismatch in an analog system. Basu *et al*. [10] proposed the first spiking-based architecture based on the ELM theory. Merkel *et al*. [11] showed an architecture similar to ours, but differed in several aspects, such as they used memristors for storage of weights and performed classification tasks in simulation. Chen *et al*. [15] developed a machine learning co-processor (MLCP) that performs spike-based computation using ELM. The MLCP encodes the ELM algorithm (with 0.4 μW power for encoding) for spike inputs in many stages, and the decoding is done separately on a microcontroller (MCU). In another work, this MLCP has configured for digital inputs for machine learning tasks [18]. The TAB framework that we proposed consists of three neuronal layers,

Manuscript received March 19, 2017; revised July 4, 2017 and August 21, 2017; accepted September 14, 2017. Date of publication October 12, 2017; date of current version March 9, 2018. This paper was recommended by Associate Editor M. Onabajo. *(Corresponding author: Chetan Singh Thakur.)*

C. S. Thakur is with the Department of Electronic Systems Engineering, Indian Institute of Science, Bengaluru 560012, India (e-mail: chetansingh84@gmail.com).

R. Wang, T. J. Hamilton, J. Tapson, and A. van Schaik are with the Biomedical Engineering and Neuroscience, the MARCS Institute, Western Sydney University, Sydney, NSW 2747, Australia

R. Etienne-Cummings is with the Department of Electrical and Computer Engineering, Johns Hopkins University, Baltimore, MD 21218 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2017.2756878

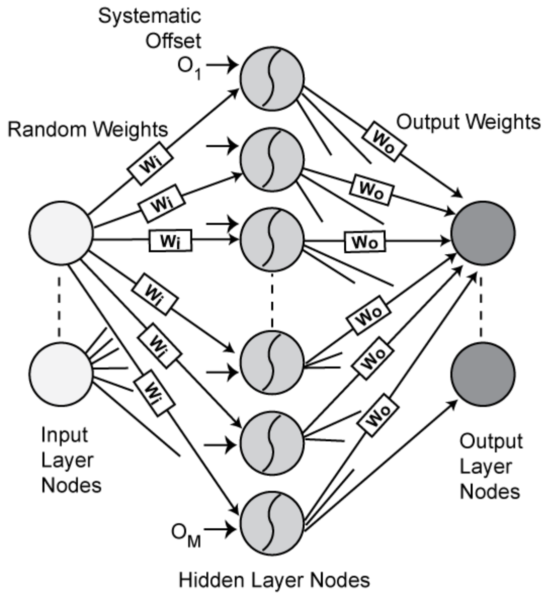

Fig. 1. **Architecture of the TAB framework.** The connections from the input layer neurons/nodes to the non-linear hidden neurons are via random weights and controllable offsets, O_1 to O_M. The hidden layer neurons are connected linearly to the outer layer neurons via trainable weights. The output neurons compute a linearly weighted sum of the hidden layer values. Adapted from [24].

namely input, hidden, and output, in a feed-forward network (Fig. 1). Fixed random weights connect the input layer neurons to a much larger number of nonlinear hidden layer neurons, thus projecting the inputs randomly and transforming them to a higher dimensional feature space. The output layer neurons compute the output weights (W_o in Fig. 1) as the product of the desired output values (Y) and the pseudoinverse of the hidden layer neuron output (H^+) [19], i.e. $W_o = H^+Y$. Matrix H is the output of all the hidden neurons for all the input training data samples. Matrix Y is the collection of the desired output vectors for the training dataset. Heterogeneity among the tuning curves of the hidden neurons is crucial for faithful encoding of information over the whole range of input stimuli. The TAB system exploits the inherent randomness (fixed-pattern transistor mismatch) to create a heterogeneous population of neurons. We cannot be certain that there would be sufficient mismatch in a particular technology until after manufacturing. This uncertainty and risk is managed by introducing a fixed and distinct systematic offset (O_i, Fig. 1), which helps to increase the diversity among the tuning curves of the hidden layer neurons.

II. VLSI Implementation of the MISO TAB

In our previous work, we built a Single Input Single Output (SISO) system using the TAB framework [20]. Here, we generalise our framework by presenting a neuromorphic co-processor. A prototype IC with two inputs and a single output, fabricated in the 130nm technology is presented as a proof-of-concept for co-processors with many more inputs. Multiple inputs allow a greater number of problem types to be solved with the MISO architecture. In general, classification and regression tasks require multiple inputs, which cannot be solved with the SISO system. The MISO system is a generalisation of the SISO system. Because of the limited die area, this Multi Input Single Output (MISO) IC was trained using offline learning, where the learnt weights were computed off the IC. The MISO system uses the same building blocks as the SISO, i.e. a hidden neuron and an output weight block (OWB). The hidden neuron model is implemented using a differential-pair circuit and performs a sigmoidal nonlinear operation on its input. The OWB block connects a hidden layer neuron with the output layer neuron using linear weights. The OWB is implemented using a current splitter circuit [20]. The MISO system also requires an additional circuit to combine the multiple inputs before passing the randomly weighted output to the hidden neuron circuit. These random weights are the result of device mismatch, which arises due to manufacturing process variations. In this section, we introduce the Weighted Average Circuit (WAC) that combines the multiple voltage inputs and discuss how it is integrated with a hidden neuron. All our circuits operate in the weak-inversion region of the transistor, which makes the TAB a very low-power system. Although transistors operating in the weak-inversion region lead to a higher mismatch, it is a desired property for our framework.

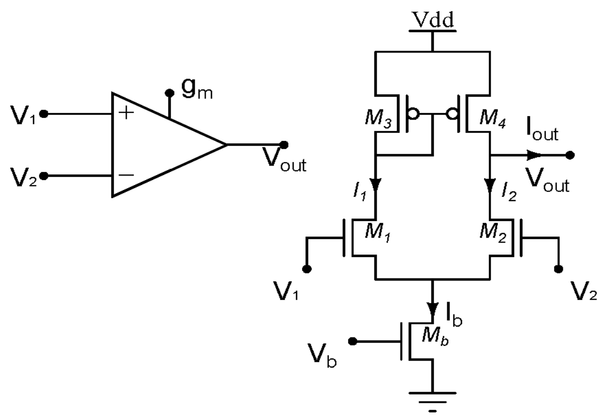

Fig. 2. **Operational Transconductance Amplifier Circuit.** Symbol (*left*) and schematic (*right*) of the circuit.

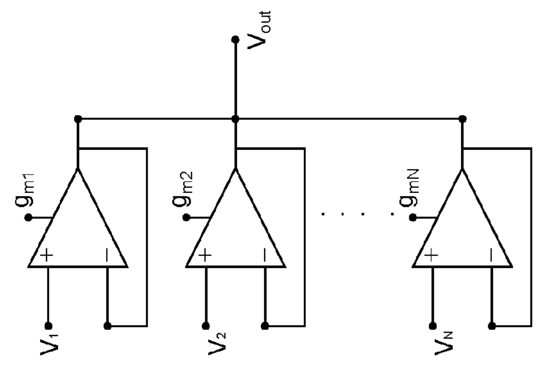

Fig. 3. **Weighted Average Circuit.** Schematic showing a Weighted Average Circuit (WAC) that is built by combining several Unity Gain Follower (UGB) circuits.

A. Weighted Average Circuit (WAC)

We used an Operational Transconductance Amplifier (OTA) to build the WAC. In Fig. 2, we show an NMOS-based OTA circuit and the symbol for this circuit.

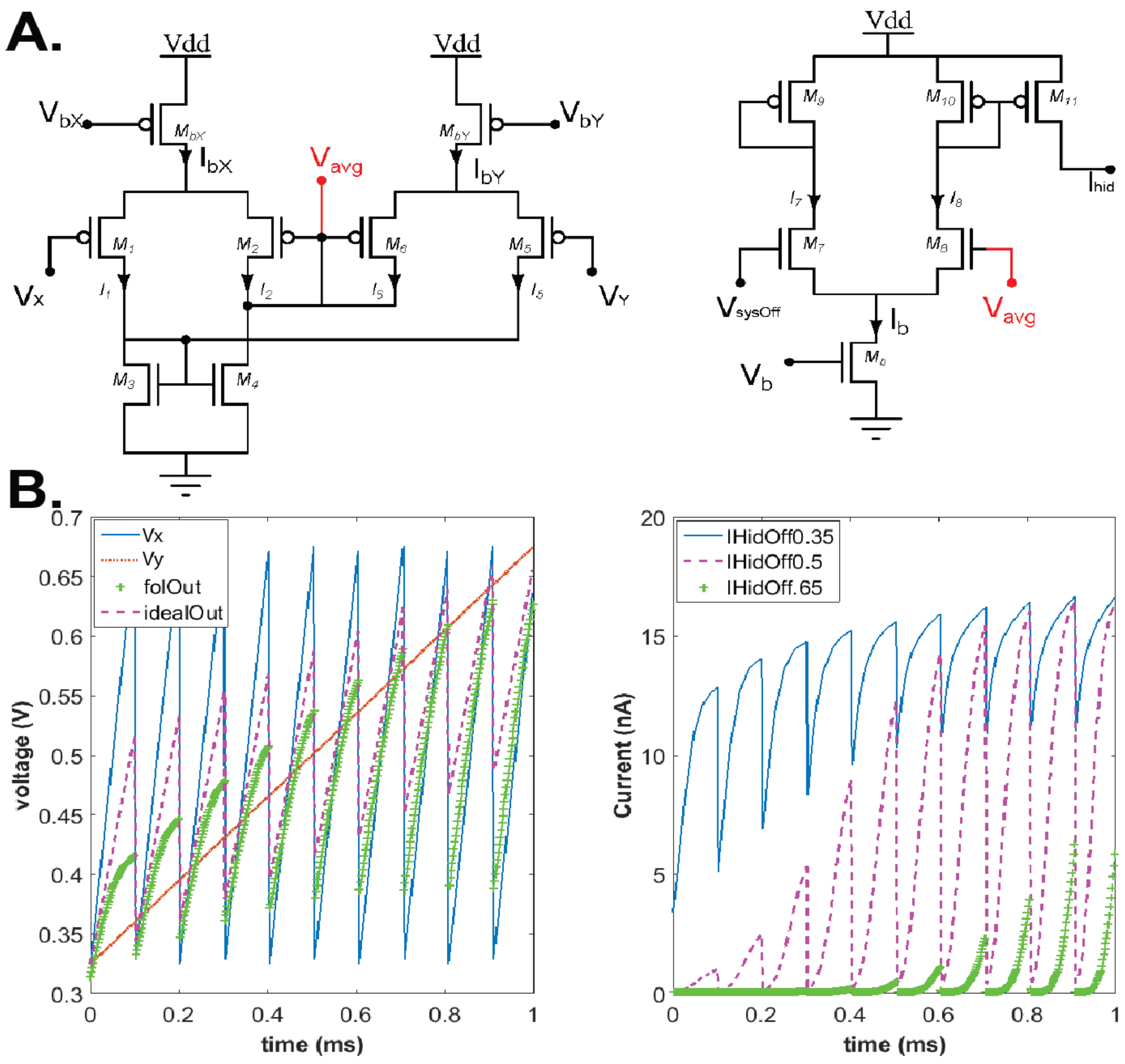

Fig. 4. **PMOS-based Weighted Average Circuit**. **(A)** Schematic of the PMOS-based WAC (*left*) and hidden neuron (*right*). **(B)** The graph on the left shows the ideal output (*magenta*−−) and the actual output (*green*+) of the WAC (V_{avg} in schematic A) for two inputs shown in red and blue. The graph on the right shows the output of the hidden neuron circuit, I_{hid}, corresponding to a reference voltage, V_{sysOff}, of 0.35 (*blue*), 0.5 (*magenta*−−), and 0.65 (*green*+).

In an OTA, the output current I_{out} is given by [21]:

$$I_{out} = I_b \tanh \frac{V_1 - V_2}{2nU_T}$$

where, I_b is the bias current, V_1, and V_2 are input voltages as shown in Fig. 2, n is the slope factor generally between 1 and 1.5, and the thermal voltage $U_T = kT/q$, where k is the Boltzmann constant, T is the temperature in Kelvin, and q is the charge of an electron. U_T is approximately 26 mV at room temperature (∼300 K).

The transconductance amplifier is biased in the weak-inversion region of the transistor, thus the current through the differential pair of transistors M_1 and M_2 is much smaller than their specific current [22], i.e.:

$$I_b \ll I_s = 2n\mu C_{ox}\frac{W}{L}U_T^2$$

where, W is the channel-width of the transistor, L is the length, μ is the mobility of the minority carriers, and C_{ox} is the gate-oxide capacitance per unit area. For a small differential voltage, the relationship between I_{out} and $(V_1 - V_2)$ is linear:

$$I_{out} \approx g_m(V_1 - V_2)$$

where, g_m is the transconductance of the amplifier, given by:

$$g_m = \frac{I_b}{2nU_T}$$

A Unity Gain Buffer (UGB) circuit can be built by connecting the negative input of the OTA to its output. This configuration behaves like a controlled conductance with respect to $(V_1 - V_{out})$, but does not draw any current from the input source. A WAC can then be built by connecting the output of several UGB circuits to compute the weighted average of several voltage inputs (Fig. 3).

In case of N inputs, the V_{out} can be expressed as [23]:

$$V_{out} = \frac{\sum_i^N g_{m_i} V_i}{\sum_i^N g_{m_i}}$$

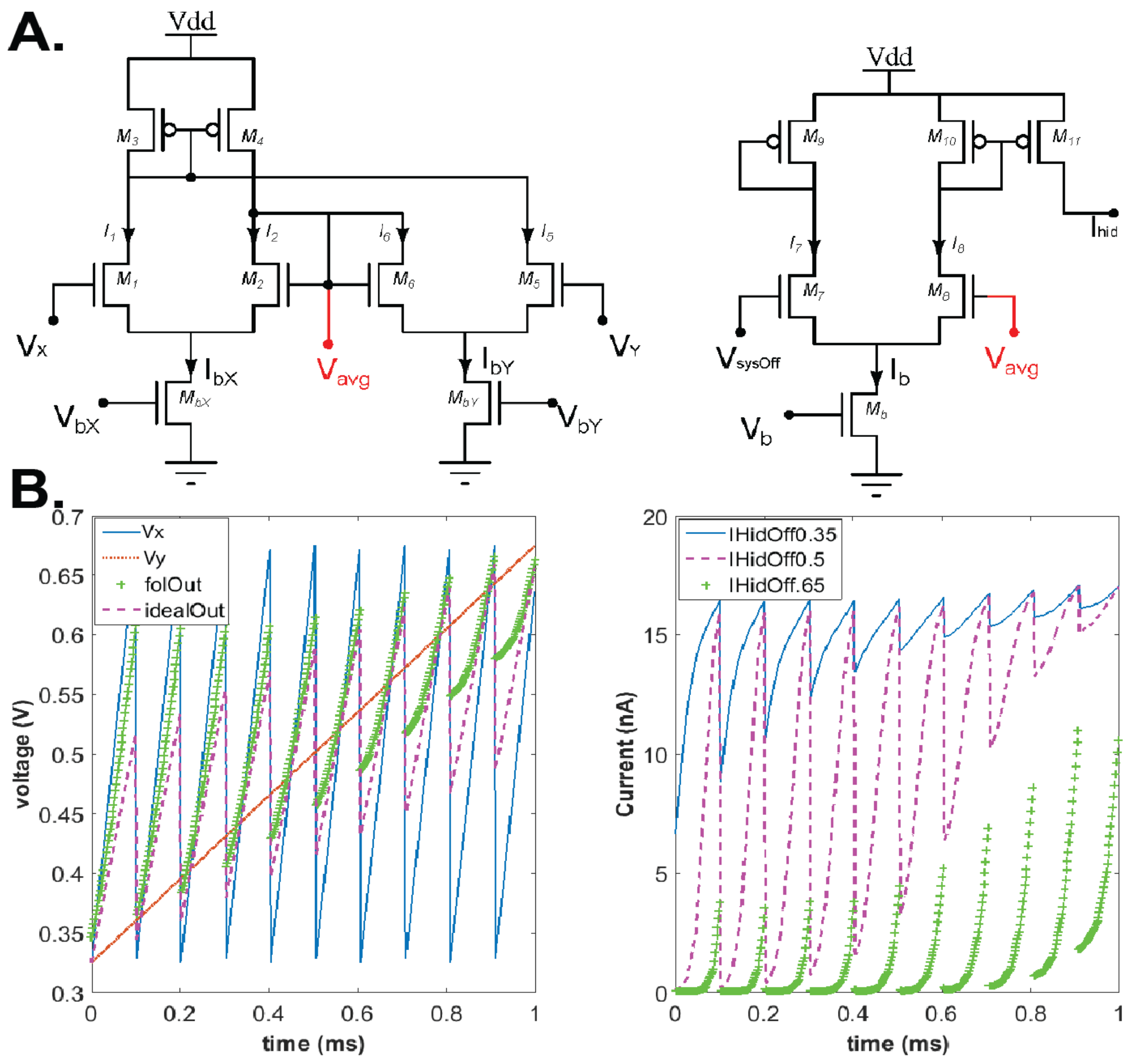

Fig. 5. **NMOS-based Weighted Average Circuit**. **(A)** Schematic of the NMOS-based WAC (*left*) and hidden neuron (*right*). **(B)** The graph on the left shows the ideal output (*magenta*−−) and the actual output (*green*+) of the WAC (V_{avg} in schematic A) for two inputs shown in red and blue. The graph on the right shows the output of the hidden neuron circuit, I_{hid}, corresponding to a reference voltage, V_{sysOff}, of 0.35 (*blue*), 0.5 (*magenta*−−), and 0.65 (*green*+).

For two inputs $V_1 \& V_2$, and their corresponding transconductances $g_{m_1} \& g_{m_2}$, the V_{out} can be expressed as:

$$V_{out} = \frac{g_{m_1} V_1 + g_{m_2} V_2}{g_{m_1} + g_{m_2}} \tag{1}$$

In the MISO TAB, we integrated the WAC with the hidden neuron, which implements the sigmoidal nonlinearity, using a differential-pair circuit, with the bias current controlled by V_b (Fig. 4A, *right* panel) and the systematic offset controlled by V_{sysoff} (Fig. 4A and Oi, Fig. 1) [20]. The WAC performs a linear-weighted average as expected by Eq. 1, but only for a small range of the input voltage because a transistor enters the weak-inversion saturation mode when the voltage between its drain and source is more than 4-5nU_T. If one of the inputs is very different, the corresponding UGB circuit tends towards saturation, which introduces nonlinear effects. Figs. 4A (*left* panel) and 5A (*left* panel) show PMOS- and NMOS-based WACs, respectively, where V_x and V_y are the inputs, V_{avg} is the weighted output, and V_{bx} and V_{by} are the biased voltages corresponding to each input. We have used a single current-mirror circuit as a load for both the UGB circuits (corresponding to each input). We can easily extend this WAC for a large number of inputs by adding extra differential pairs (one for each input) and keeping a single current-mirror load, thus saving significant silicon area. The output of the WAC, V_{avg}, is connected as an input to a hidden neuron (*right* panel in Figs. 4A and 5A) modelled using a differential-pair circuit. The systematic offset, V_{sysoff}, is another input connected to the hidden neuron. The output current, I_{hid}, of the hidden neuron is connected to the OWB. The circuit simulations in Figs. 4B and 5B (*left* panel) show that when g_{m_1} and g_{m_2} are equal, the output of the WAC (*green* curve) is not the same as the ideal output (*magenta* curve). For the input range used, some nonlinear effects are observed in the output voltage in the WAC. These nonlinear effects are much more prominent in the lower and higher input ranges for the PMOS- and NMOS-based WACs, respectively (Figs. 4 and 5). In the TAB, there is a WAC corresponding to

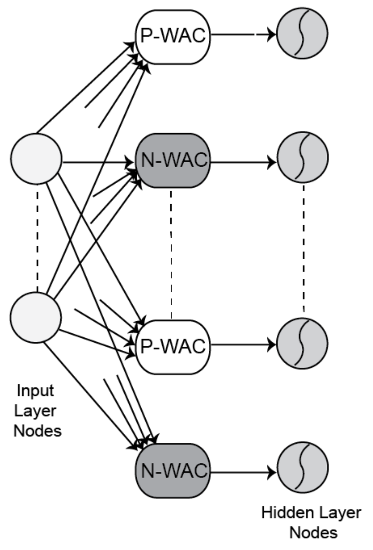

Fig. 6. Schematic showing the first two layers of a multi-input TAB framework, with alternate P-type and N-type WACs.

each hidden neuron. To minimise nonlinear effects when combining multiple inputs, we have used both the types of WACs, PMOS and NMOS, alternately in the MISO TAB (Fig. 6), as they have nonlinear effects in the opposite directions of the input. In order to keep the nonlinearity balanced, we have used an equal number of each type of WAC. For example, some of the regression functions might have nonlinearity in only one direction, and in such cases, only one type of WAC may not be sufficient to realize that function. We have thus used 50 PMOS- and 50 NMOS-based WACs for a total of 100 hidden neurons in our co-processor. The nonlinear effect in the WAC can also be minimised by using a smaller input range.

The WACs in Figs. 4A and 5A show that two inputs, V_x and V_y, are passed through the WAC, which combines these inputs as a weighted sum, and the weights depend on the g_m corresponding to each input. The voltages V_{bx} and V_{by} at the transistors M_{bx} and M_{by}, respectively, set the bias current ($\sim$ few nanoamperes), which determines the corresponding g_m for each UGB circuit (Figs. 4A and 5A). In the TAB, each UGB circuit in the WAC will experience a distinct g_m, even for the same bias voltage, due to the variability in $M_b$caused by process variations. These random values of g_m act as random weights in the TAB framework. The mismatch in the transistors, $\mathrm{M_{bx}}/\mathrm{M_{by}}$, are modelled as a log-normal distribution because of the exponential relationship between current and voltage that results in the random weights. The size (W/L) of all transistors in circuits shown in Figs. 2, 4, and 5 was set at 280nm/280nm to ease the layout.

III. Software Model of MISO TAB

Here, we describe the software simulations of the MISO TAB for a two input, single output configuration. In our software model, we have used sigmoidal nonlinearity because this nonlinearity is used later for the hardware implementation. We have also modelled in software the monotonic nonlinearity of the WACs as observed in the circuit simulations (Fig. 4). We presented the training data to the network, with each training pair consisting of two inputs and an output. Each input training value was multiplied by the random weights for each hidden neuron and projected randomly to 100 hidden neurons (Fig. 1). We collected the response of the hidden neurons for every input data point. The output weights were computed as the product of the desired output values and the pseudoinverse of the hidden neurons' output. In the testing phase, we presented the test input to the network and obtained a 'predicted' output.

The following constraints were imposed in the software simulations:

- All transistors in the TAB system were biased in the weak-inversion (sub-threshold) region of transistors. This limits the input range to a few hundred millivolts because a transistor enters the weak-inversion saturation mode when the voltage between its drain and source is equal to or more than 4-5nU_T $\sim=$ 100–200 mV (n could be in the range of 1–1.5).
- The TAB only supports positive input voltages on the IC. We can appropriately scale and offset the external inputs to support the input range of the TAB.
- Similarly, the target functions also need proper scaling and offsetting (for negative values of the outputs).
- We modelled the values of random weights and random offsets based on the characterisation of the TAB SISO IC reported previously [20].

We verified the performance of the MISO TAB by testing its ability to learn functions of varying complexities–low, medium, and high. A high-complexity function such as the *sinc* function ($sinc(\pi(x^2+y^2))$) has more inflexions or higher spatial frequency components. On the other hand, a low-complexity function such as the *square* function (x^2+y^2) consists of lower spatial frequency components. For a medium-complexity function, we have chosen $\sin\left(\pi x^2\right)+cos(\pi y^2)$, which we refer to as the *trigonometric square* function. In order to determine the optimal number of output weight bits, *Wbits*, and hidden neurons required for each kind of function, we varied these parameters and analysed the errors thus obtained for the *square*, *sinc*, and the *trigonometric square* functions. Here, error is defined as the RMS value of the difference between the learnt and target functions, divided by the RMS value of the target function. As shown in Fig. 7, 12 bits and 100 hidden neurons were sufficient to represent the *square* and*trigonometric square* functions to achieve an error less than 0.1, but a higher number of output weight bits and hidden neurons were required for the *sinc* function. As shown in Fig. 8, the TAB MISO system with 100 neurons and 12-bit output weight was able to learn the *square* and *trigonometric square* functions and performed these regression tasks with marginal errors of 0.05 and 0.07, respectively. The random input weights vary according to a log-normal distribution owing to the exponential relationship between the voltage and the current of a transistor in the weak-inversion region. We have characterised (average of 10 trials) the performance of the *square* and *trigonometric square* functions for various values of the standard deviations as shown in Fig. 9.

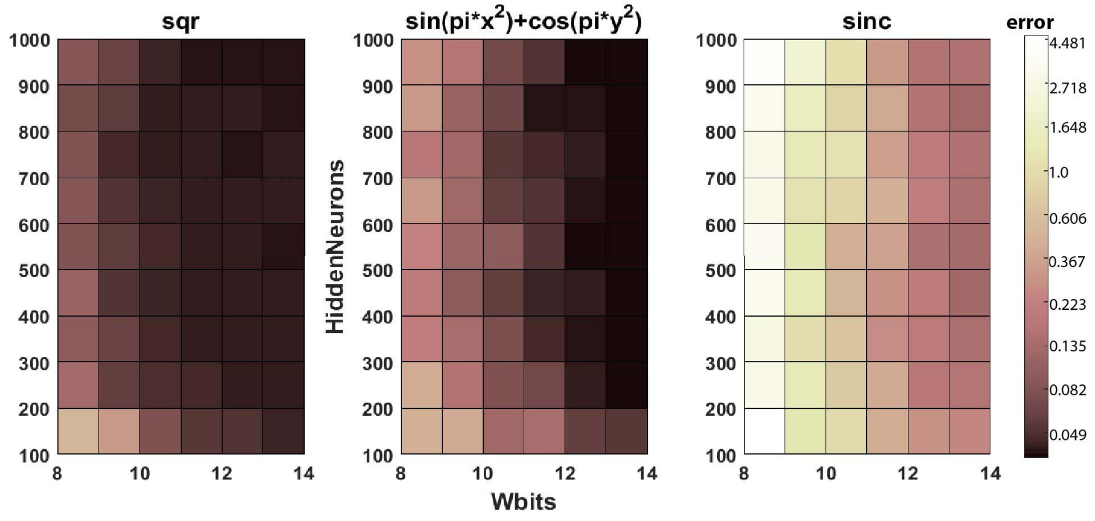

Fig. 7. **Characterisation of error in the MISO TAB**. The numbers of hidden neurons and output weight bits (*Wbits*) were varied and the error was calculated for the functions: (A) *Square*, (B) *Trignometric Square*, and (C) *Sinc*.

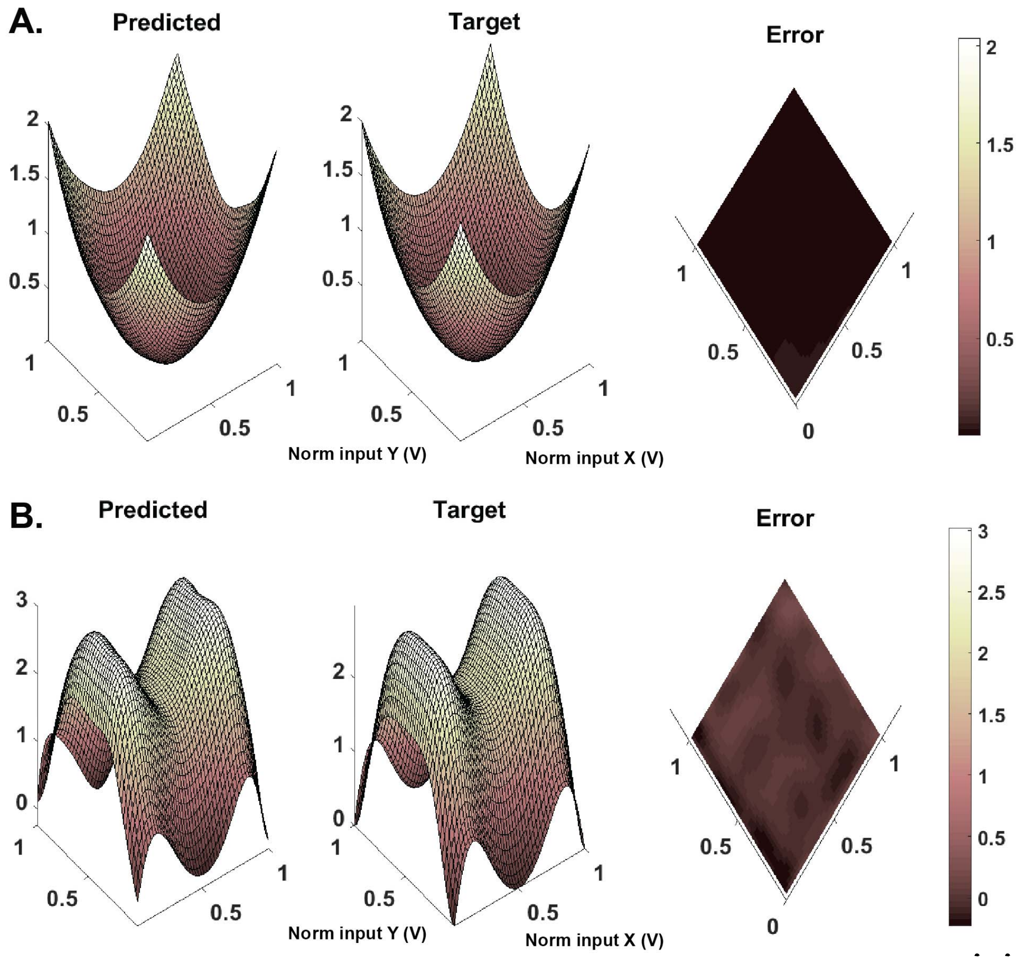

Fig. 8. **Learning ability of the MISO TAB using 100 hidden neurons and 13-bit output weight resolution. (A)** *Square* function $(x^2 + y^2)$. **(B)** *Trigonometric Square* function $(\sin(\pi x^2) + \cos(\pi y^2))$.

IV. IC Results

A. Neuron Characterisation in the Co-Processor

Our co-processor IC has 100 neuron blocks, each of which includes a WAC, a hidden neuron, an OWB, and 12-bit shift registers. Table 1 summarises the system-level features of this co-processor IC. In the TAB framework, voltage and current represent the physical variables at the input and output layer neurons, respectively. All transistors in our system are biased in the weak-inversion region to create a low-power system.

TABLE I
FEATURES OF THE TAB MISO IC

Technology	IBM 130nm
Supply voltage	1.2 V
Total number of neuron blocks	100
Total power without output current amplifier	1.1 μW
Total power	15.6 μW
Time constant of the TAB	5.2 μS
Area of each circuit per block and IC (μm × μm)	
Hidden neuron	5 × 4.5
Weighted Average Circuit (P- & N-type)	5.5 × 4.5
Output weight circuit	36 × 5
Shift registers (11 bits weight + Sign bit)	43 × 14
IC dimension	2222 × 58.5

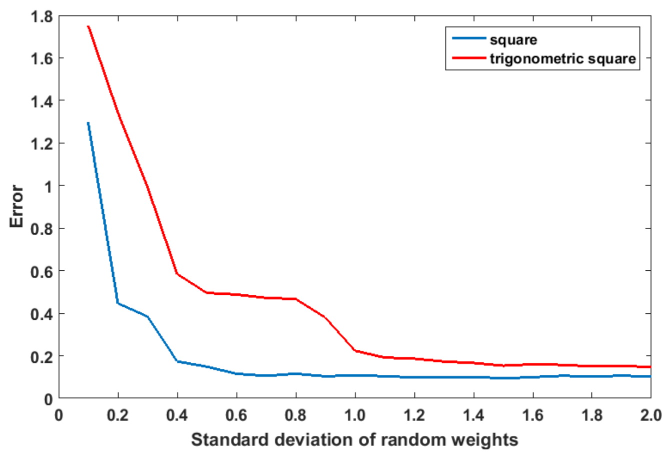

Fig. 9. Characterisation (average of 10 trials) of the performance of the *square* and *trigonometric square* functions for various values of the standard deviations of the random weights.

Thus, the current at the output neurons is in the range of hundreds of nanoamperes. For measurement purposes, we have used a current-gain circuit (100×) consisting of two sets of current mirrors, each with a gain of 10, to amplify the final output current. This circuit may be unnecessary in actual applications.

We characterised the tuning curve of each neuron to analyse the mismatch and differences between the tuning curves in the absence of systematic offset, by connecting the V_{sysOff} node (Figs. 4 and 5) of each hidden neuron to the same voltage. We used a bias voltage of 0.15 V and 0.9 V for all the N-type and P-type WACs, respectively. As shown in Fig. 10B, the neuronal tuning curves were heterogeneous due to random device mismatch and process variations in the fabrication. The output weights of all the hidden neurons were connected serially as a long chain of shift registers. The output current of each hidden neuron was probed indirectly through the 'OUT' port of the IC sequentially by writing all ones to the corresponding output weight and setting all other neurons' output weights to zero, thus allowing current only from the selected hidden neuron. Then, we provided two ramp inputs to the IC such that they covered all possible combinations in the input space, and measured the current at the output port. A sample tuning curve thus obtained is shown in Fig. 10A. We collected the tuning curves for all neurons (Fig. 10B) and computed the basis functions for these tuning curves by applying the singular value decomposition (SVD) technique on the correlation matrix of the tuning curves. The singular values for all but the first few basis functions were very small, implying that their contribution in encoding of the inputs was negligible. Thus, we have only shown the first sixteen basis functions in Fig. 10D. The shape of a basis function is indicative of the kind of functions that could be supported or decoded by the corresponding neuron population. High spatial frequencies in the basis functions of the framework suggest a high capability for encoding and learning complex functions.

In Fig. 10C, we show the variations in tuning curves obtained by changing the bias voltage for the P-type WAC in the test cell, while keeping the systematic offset of all the hidden neurons the same and constant. The WAC contains two followers corresponding to each input. In the WAC, the bias voltage of one of the followers was varied from 870 mV to 990 mV, while keeping the other follower at 900 mV (Fig. 4). Our results show that by controlling the bias voltage of the WAC, different tuning curves can be obtained, and thus the heterogeneity among the population of neurons can be increased.

B. Learning in the MISO TAB

In this section, we use offline learning to show the ability of the TAB MISO IC to learn by exploiting device mismatch. We collected all the 100 tuning curves, as explained in the previous section, and calculated the output weights external to the IC in the range $(-1,1)$. In Fig. 11, we show two simple learning tasks to demonstrate the learning capability of the IC. Using synthetic data, we trained the MISO IC for the 'XOR' problem, a standard nonlinear classification problem in the machine learning community, with an accuracy of 98.2% (Fig. 11A). Similarly, we trained the IC to classify three different types of synthetic data clusters with 96.5% accuracy (Fig. 11B).

In order to assess the learning capability of the MISO IC for complex regression functions, we used the measured tuning curves obtained from the IC without any systematic offset, but with off-chip implementation of the

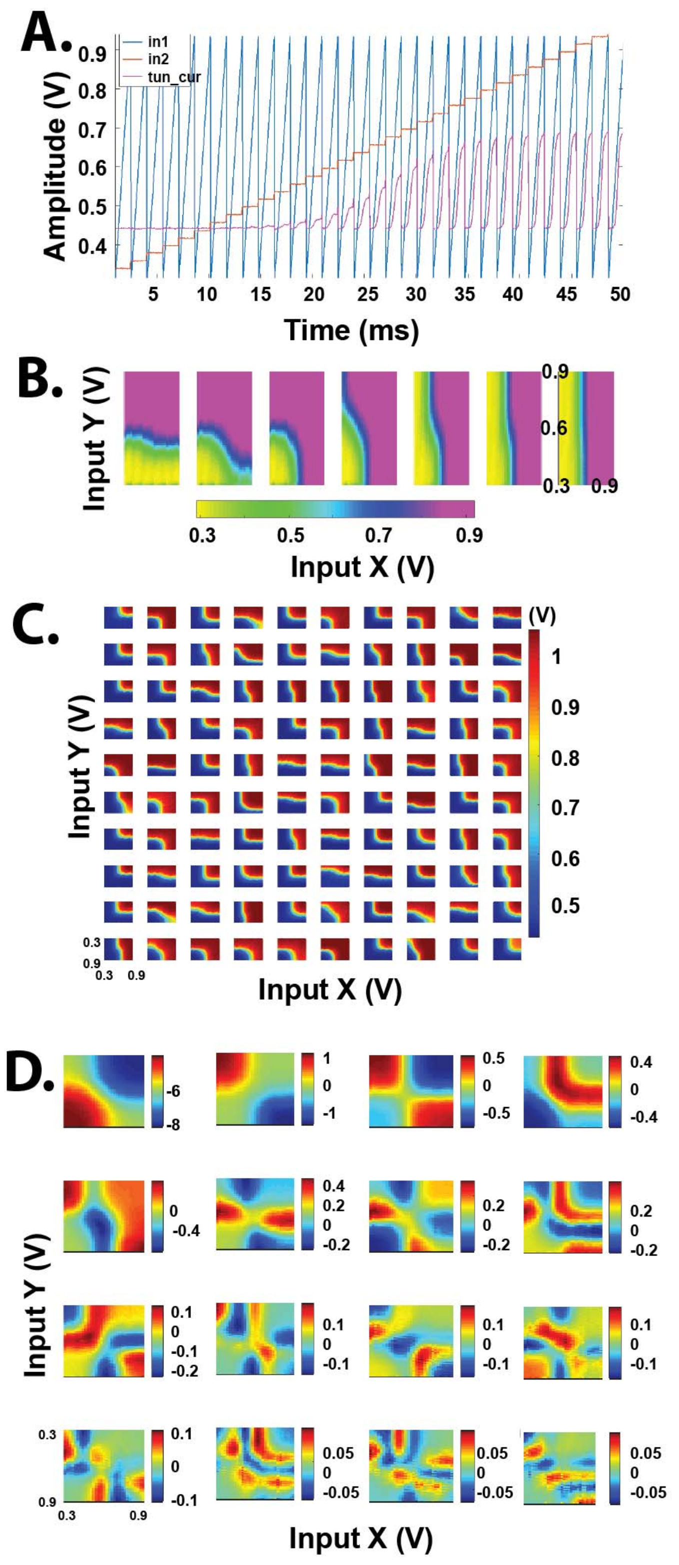

Fig. 10. **Characterisations of the hidden neurons** **(A)** Tuning curve of a neuron (magenta) as a function of two inputs (blue and red). **(B)** Variation in the tuning curves of the test cell by varying g_m of the WAC. **(C)** Tuning curves of 100 hidden neurons in three-dimensional representation. **(D)** Top 16 basis functions obtained from the tuning curves of 100 hidden neurons. Colour maps show the relative magnitude.

decoding component of the TAB, for reasons explained below. In Fig. 12, we show the training results of the TAB MISO IC for the *square* and *trigonometric square* tasks with an RMS error of 0.19 and 0.21, respectively, with respect to the target RMS. The performance of the system could be improved further by increasing the heterogeneity of the tuning curve population by using systematic offsets [24] Our results demonstrate the encoding capacity of the system and show that complex functions can be learnt with the TAB system.

We encountered two problems while configuring the MISO IC for learning: (1) Occasionally, a few bits in the shift registers did not set in the registers correctly, due to the insufficient driving capability of the clock buffers in the path. (2) The relationship between the digital weight and the current gain is nonlinear, which would be insignificant if we used the IC in a training loop [25]. However, the shift register for the weights was unreliable in this implementation, and thus training with the IC in the loop was impossible. Owing to these problems, it was difficult to train the MISO IC correctly for complex regression tasks. Therefore, we used the measured turning curves with ideal off-chip weights to demonstrate learning. These problems encountered with the test IC could be easily fixed, as there were no conceptual issues related to our co-processor architecture. Furthermore, the accuracy can be improved by using a larger number of hidden neurons, but we chose to keep only 100 hidden neurons due to the restrictions of the size of the die area.

V. Building Dynamical Systems Using TAB

The TAB framework developed here is a type of feed-forward neural network. We could easily extend the capability of the TAB by connecting multiple TABs together with feed-back connections to build dynamical systems. Our future aim is to build a general purpose IC that could be configured to build any dynamical system by connecting multiple TABs in an appropriate fashion. Such dynamical systems could be re-trained and applied to engineering problems. Eliasmith *et al.* have modelled various neurobiological dynamical systems [26] using their NEF framework with spiking neurons [27], which is similar to the TAB. Here, we describe a similar approach to build a sample dynamical system, a controlled oscillator, using TABs in software simulations.

$$x_1(t) = \int (A_1 x_1(t) + B_1 f(x_2, u))$$
$$x_2(t) = \int (A_2 x_2(t) + B_2 f(x_1, u))$$
$$y(t) = C x_1(t) + D u(t)$$

where, $x_1(t)$ and $x_2(t)$ are the state variables of the dynamical system; $u(t)$ is the input or control variable; $y(t)$ is the output; $A_1, A_2, B_1, B_2, C,$ and D are time invariant coefficients; $f(x_2, u) = (x_2 \times u)$; $f(x_1, u) = (x_1 \times u)$, and are learnt using the TAB (two inputs and single output). For the controlled oscillator, the values of the coefficients are:

$$A_1 = 1;\ A_2 = 1;\ B_1 = 1;\ B_2 = -1;\ C = 1;\ D = 0$$

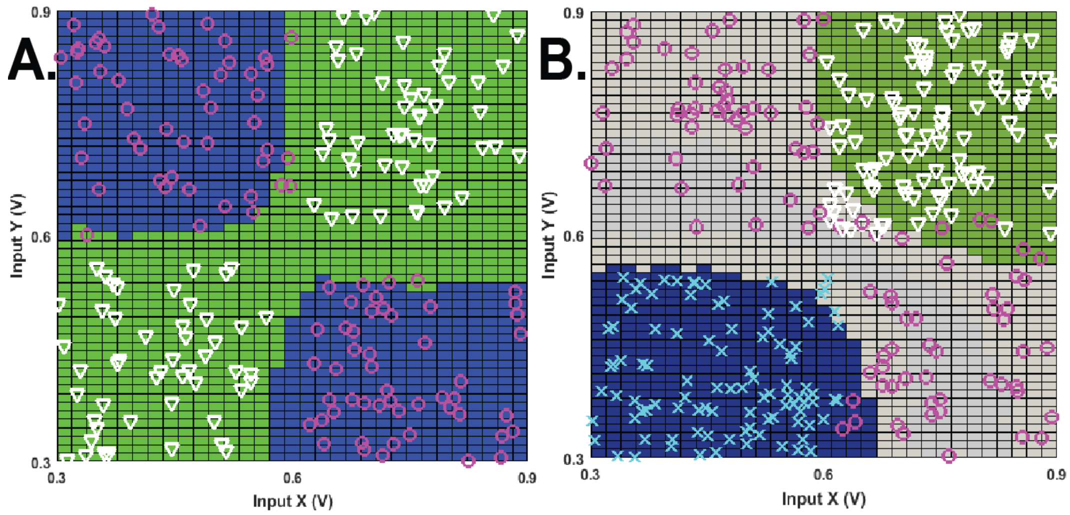

Fig. 11. **Training of the TAB MISO IC for classification tasks.** **(A)** XOR problem, i.e. separating two data clusters (*magenta* ‘o’ and *white* ‘v’) by creating a nonlinear classification boundary. **(B)** Classifying three different data clusters (*magenta* ‘o’, *white* ‘v’, and *cyan* ‘x’).

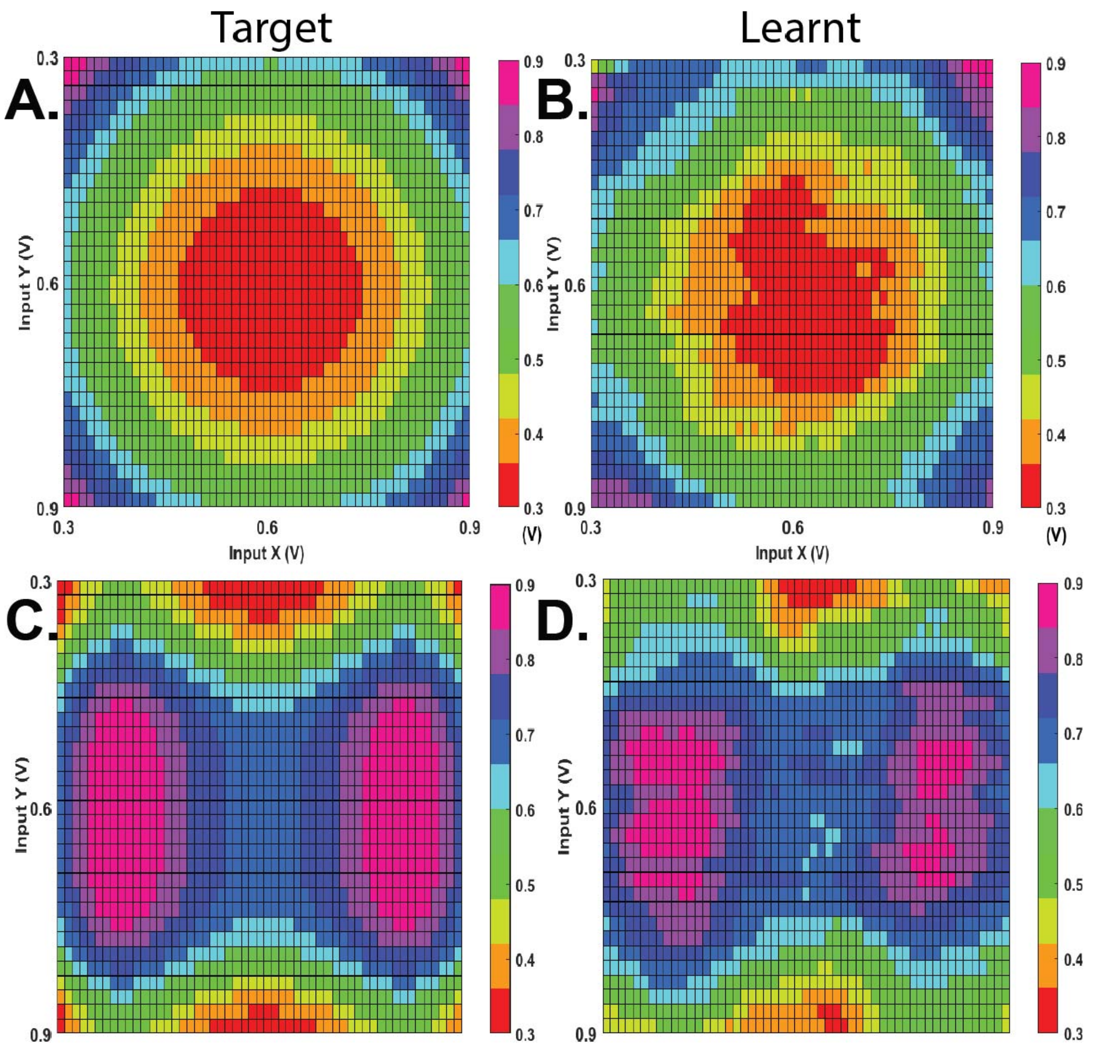

Fig. 12. **Training of the TAB MISO IC for regression tasks**. Training was done using measured tuning curves with decoding done in software, for the functions: **(A)** Square ($x^2 + y^2$), and **(C)** Trigonometric Square ($\sin(\pi x^2) + \cos(\pi y^2)$).

The integrator can be built using off-chip RC circuits. The simulation results of this system are shown in Fig. 13. The frequency of the oscillator is shown to change as a function of the control variable $u(t)$. Similarly, any other dynamical system can be modelled using the TABs.

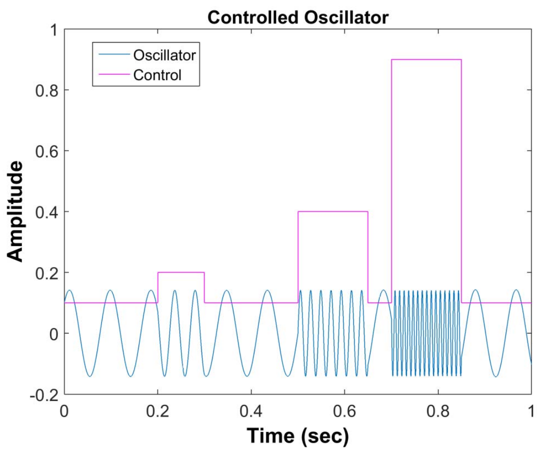

Fig. 13. **A controlled oscillator.** The frequency of the oscillator (*blue*) changes as a function of control input, *u* (*pink*).

VI. Conclusions

We have extended our neuromorphic TAB architecture to generalise it for the multi-input scenario. We presented measurement results of our prototype IC designed in the 130nm technology for the MISO configuration of the TAB system. We also showed the learning capability of our co-processor for various regression and classification tasks. Our TAB architecture exploits random device mismatch (fixed-pattern mismatch) and variability in the fabrication process. The TAB also incorporates systematic offset as a failsafe method to spread the tuning curves of the neurons. Systematic offset may be required when there is insufficient random variation among transistors to produce a distinct tuning curve for each neuron.

Our co-processor performs computation in the analogue domain, and the output weights are stored in the shift registers. We have implemented our framework in the analogue domain, which is superior to digital implementations [28]–[32]. For example, summation in an analogue circuit is computed simply using Kirchhoff's current law by connecting the common output line. Similarly, multiplication is implemented using the output weight circuits with a few transistors, while a digital implementation requires several thousands of transistors for the same computations. Our system also offers very low power consumption in the range of a few μW (Table 1). In our TAB system, the inputs are randomly projected from their original input dimensionality to a nonlinear hidden layer of neurons of a much higher dimensionality. The temperature variation will not affect this random projection. However, it will affect the learning of the output weights if they have been learnt at a particular temperature. Therefore, in future work, we will incorporate temperature as a random variable and the output weights will be trained considering the range of temperature. Additionally, the current implementation of the neurons could face problems owing to the dynamic range of the input for some applications; however, given the simplicity of the design, it is a very simple matter to replace the simple differential pair OTAs with wide-swing OTAs [33] depending on the applications.

Our system performs classification directly on analogue inputs without needing to digitise them. This approach could be easily extended to a larger number of inputs. A larger TAB could, for instance, be used in a smart camera for recognising features such as faces or text. We can train the TAB to implement the analogue equivalent of a Finite Impulse Response (FIR) filter. By feeding delayed versions of the output signal back to some of the input nodes, as shown in Fig. 12 for a dynamical system, we can also implement the analogue equivalent of an Infinite Impulse Response (IIR) filter. Furthermore, these filters can be re-trained if the desired filter function changes.

References

[1] G. E. Moore, "Cramming more components onto integrated circuits," *IEEE Solid-State Circuits Newslett.*, vol. 20, no. 3, pp. 33–35, Sep. 2006, reprinted from *Electronics*, vol. 38, no. 8, Apr. 1965, pp. 114 ff.
[2] G. Indiveri *et al.*, "Neuromorphic silicon neuron circuits," *Frontiers Neurosci.*, vol. 5, no. 73, pp. 1–23, 2011.
[3] C. S. Thakur, T. J. Hamilton, J. Tapson, A. van Schaik, and R. F. Lyon, "FPGA implementation of the CAR model of the cochlea," in *Proc. IEEE Int. Symp. Circuits Syst.*, Jun. 2014, pp. 1853–1856.
[4] S.-C. Liu and T. Delbruck, "Neuromorphic sensory systems," *Current Opinion Neurobiol.*, vol. 20, no. 3, pp. 288–295, 2010.
[5] T. J. Sejnowski, "Neural populations revealed," *Nature*, vol. 332, p. 308, Mar. 1988.
[6] M. I. Chelaru and V. Dragoi, "Efficient coding in heterogeneous neuronal populations," *Proc. Nat. Acad. Sci. USA*, vol. 105, no. 42, pp. 16344–16349, Oct. 2008.
[7] M. Rigotti *et al.*, "The importance of mixed selectivity in complex cognitive tasks," *Nature*, vol. 497, no. 7451, pp. 585–590, May 2013.
[8] A. S. Ecker, P. Berens, A. S. Tolias, and M. Bethge, "The effect of noise correlations in populations of diversely tuned neurons," *J. Neurosci.*, vol. 31, no. 40, pp. 14272–14283, 2011.
[9] K. Cameron, V. Boonsobhak, A. Murray, and D. Renshaw, "Spike timing dependent plasticity (STDP) can ameliorate process variations in neuromorphic VLSI," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1626–1637, Nov. 2005.
[10] A. Basu, S. Shuo, H. Zhou, M. H. Lim, and G.-B. Huang, "Silicon spiking neurons for hardware implementation of extreme learning machines," *Neurocomputing*, vol. 102, pp. 125–134, Feb. 2013.
[11] C. Merkel and D. Kudithipudi, "Neuromemristive extreme learning machines for pattern classification," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI*, Jul. 2014, pp. 77–82.
[12] T. J. Hamilton, S. Afshar, A. van Schaik, and J. Tapson, "Stochastic electronics: A neuro-inspired design paradigm for integrated circuits," *Proc. IEEE*, vol. 102, no. 5, pp. 843–859, May 2014.
[13] T. C. Stewart and C. Eliasmith, "Large-scale synthesis of functional spiking neural circuits," *Proc. IEEE*, vol. 102, no. 5, pp. 881–898, May 2014.
[14] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, nos. 1–3, pp. 489–501, 2006.
[15] Y. Chen, E. Yao, and A. Basu, "A 128 channel 290 GMACs/W machine learning based co-processor for intention decoding in brain machine interfaces," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2015, pp. 3004–3007.
[16] O. Richter, R. F. Reinhart, S. Nease, J. Steil, and E. Chicca, "Device mismatch in a neuromorphic system implements random features for regression," in *Proc. IEEE Biomed. Circuits Syst. Conf. (BioCAS)*, Oct. 2015, pp. 1–4.
[17] F. Corradi, C. Eliasmith, and G. Indiveri, "Mapping arbitrary mathematical functions and dynamical systems to neuromorphic VLSI circuits for spike-based neural computation," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Jun. 2014, pp. 269–272.
[18] E. Yao and A. Basu, "VLSI extreme learning machine: A design space exploration," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 1, pp. 60–74, Jan. 2017.
[19] J. Tapson and A. van Schaik, "Learning the pseudoinverse solution to network weights," *Neural Netw.*, vol. 45, pp. 94–100, Sep. 2013.
[20] C. S. Thakur, R. Wang, T. J. Hamilton, J. Tapson, and A. V. Schaik, "A low power trainable neuromorphic integrated circuit that is tolerant to device mismatch," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 63, no. 2, pp. 211–221, Feb. 2016.

[21] S. C. Liu, *Analog VLSI: Circuits and Principles*. Cambridge, MA, USA: MIT Press, 2002.

[22] E. A. Vittoz, "Analog VLSI signal processing: Why, where, and how?" *J. VLSI Signal Process. Syst. Signal, Image Video Technol.*, vol. 8, no. 1, pp. 27–44, Feb. 1994.

[23] E. A. Vittoz, "Analog VLSI implementation of neural networks," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 1990, pp. 2524–2527.

[24] C. S. Thakur, T. J. Hamilton, R. Wang, J. Tapson, and A. van Schaik, "A neuromorphic hardware framework based on population coding," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2015, pp. 1–8.

[25] C. S. Thakur, R. Wang, S. Afshar, T. J. Hamilton, J. Tapson, and A. van Schaik. (May 2015). "An online learning algorithm for neuromorphic hardware implementation." [Online]. Available: https://arxiv.org/abs/1505.02495

[26] C. Eliasmith, "A unified approach to building and controlling spiking attractor networks," *Neural Comput.*, vol. 17, no. 6, pp. 1276–1314, Jun. 2005.

[27] C. Eliasmith and C. Anderson, *Neural Engineering: Computation, Representation, and Dynamics in Neurobiological Systems*. Boston, MA, USA: MIT Press., 2003.

[28] R. Gadea, J. Cerdá, F. Ballester, and A. Macholí, "Artificial neural network implementation on a single FPGA of a pipelined on-line back-propagation," in *Proc. 13th Int. Symp. Syst. Synth.*, 2000, pp. 225–230.

[29] M. Bahoura and C.-W. Park, "FPGA-implementation of high-speed MLP neural network," in *Proc. 18th IEEE Int. Conf. Electron., Circuits, Syst.*, Dec. 2011, pp. 426–429.

[30] S. Himavathi, D. Anitha, and A. Muthuramalingam, "Feedforward neural network implementation in FPGA using layer multiplexing for effective resource utilization," *IEEE Trans. Neural Netw.*, vol. 18, no. 3, pp. 880–888, May 2007.

[31] A. R. Omondi and J. C. Rajapakse, *FPGA Implementations of Neural Networks*. Dordrecht, The Netherland: Springer, 2006.

[32] R. Wang, C. S. Thakur, T. J. Hamilton, J. Tapson, and A. van Schaik. (Jul. 2015). "A neuromorphic hardware architecture using the neural engineering framework for pattern recognition." [Online]. Available: https://arxiv.org/abs/1507.05695

[33] R. Sarpeshkar, R. F. Lyon, and C. Mead, "A low-power wide-linear-range transconductance amplifier," *Analog Integr. Circuits Signal Process.*, vol. 13, nos. 1–2, pp. 123–151, 1997.

Chetan Singh Thakur (M'14) received the M.Tech. degree in biomedical engineering from the Indian Institute of Technology, Bombay, India, in 2007, and the Ph.D. degree from the MARCS Institute, Western Sydney University, Australia, in 2016. He was a Senior Integrated Circuit Design Engineer in the area of mobile processor with Texas Instruments. He holds a post-doctoral at Johns Hopkins University. He is currently an Assistant Professor with the Indian Institute of Science, India. His research interests include neuromorphic engineering, stochastic electronics and computational neuroscience.

Runchun Wang (M'13) received the M.Sc. degree in electrical engineering from the Shanghai Jiaotong University, Shanghai, China, in 2008, and the Ph.D. degree in neuromorphic engineering from Western Sydney University, Sydney, Australia, in 2013. He is currently a Post-Doctoral Fellow with Biomedical Engineering and Neuroscience research program. His research focuses on neuromorphic engineering, mixed-signal/analog VLSI design, ASIC/SoC/FPGA design, computational neuroscience, deep neural network, machine leaning, cognition systems, and signal processing.

Tara Julia Hamilton (S'97–M'00) received the B.E. degree (|Hons.) in electrical engineering and the B.Com. degree from The University of Sydney, Australia, in 2001, the M.Eng.Sc. degree in biomedical engineering from the University of New South Wales, Sydney, in 2003, and the Ph.D. degree from The University of Sydney in 2008. She is currently a Senior Research Lecturer in bioelectronics and neuroscience with the MARCS Institute, Western Sydney University, Australia. Her current research interests include neuromorphic engineering, mixed-signal integrated circuit design, and biomedical engineering.

Ralph Etienne-Cummings (F'13) received the B.S. degree in physics from Lincoln University, Lincoln, PA, USA, in 1988, and the M.S.E.E. and Ph.D. degrees in electrical engineering from the University of Pennsylvania, Philadelphia, PA, USA, in 1991 and 1994, respectively. He is currently a Professor of electrical and computer engineering, and computer science with Johns Hopkins University, Baltimore, MD, USA. He was the Founding Director of the Institute of Neuromorphic Engineering. He has authored over 200 peer-reviewed articles and holds numerous patents. He was elected as a member of Circuits and Systems (CAS) Board of Governors. He also serves on numerous editorial boards. He was a recipient of the NSFs Career and Office of Naval Research Young Investigator Program Awards. He was a Visiting African Fellow with the University of Cape Town, the Fulbright Fellowship Grantee, the Eminent Visiting Scholar at the University of Western Sydney. He received numerous publication awards, including the 2012 Most Outstanding Paper of the IEEE TRANSACTIONS ON NEURAL SYSTEMS AND REHABILITATION ENGINEERING. He was recently recognized as a Science Maker and an African American history archive. He has served as the Chairman of various IEEE CAS Technical Committees.

Jonathan Tapson (M'05) the B.Sc. degree in physics, the B.Sc. degree in electrical engineering, and the Ph.D. degree in engineering from the University of Cape Town. He is currently a Professor with Western Sydney University, where he joined in 2011, and has been the Head of electrical engineering with the University of Cape Town, South Africa.

His research interests are in bio-inspired sensors and systems. With his co-authors, he is currently involved on a major program of analog and mixed signal IC design in the area of stochastic electronics. He is a former President of the South African Council on Computation and Automation, and a fellow of the South African Academy of Engineering.

André van Schaik (M'00–SM'02–F'14) received the M.Sc. degree in electrical engineering from the University of Twente, Enschede, The Netherlands, in 1990, and the Ph.D. degree in electrical engineering from the Swiss Federal Institute of Technology, Lausanne, Switzerland, in 1998. He is currently a Professor of bioelectronics and neuroscience with the MARCS Institute, Western Sydney University, where he leads the biomedical engineering and neuroscience program. He has authored over 200 publications. He is an inventor of over 30 patents. His research focuses on three main areas: neuromorphic engineering, bioelectronics, and neuroscience. He is a Founder of three start-up companies.