Journal of Circuits, Systems, and Computers
Vol. 26, No. 11 (2017) 1750183 (19 pages)

DOI: 10.1142/S0218126617501833

Integrated Circuit with Memristor Emulator Array and Neuron Circuits for Biologically Inspired Neuromorphic Pattern Recognition*

Rajeev Ranjan[†,§], Pablo Mendoza Ponce[†,‡], Wolf Lukas Hellweg[†], Alexandros Kyrmanidis[†], Lait Abu Saleh[†], Dietmar Schroeder[†] and Wolfgang H. Krautschneider[†]

[†]*Institute of Nano- and Medical Electronics, Hamburg University of Technology, Eissendorfer Str. 38, 21073 Hamburg, Germany*

[‡]*Escuela de Ingeniería Electrónica, Tecnológico de Costa Rica, Cartago, Costa Rica*
[§]*rajeev.ranjan@tuhh.de*

Received 1 November 2016
Accepted 7 March 2017
Published 27 April 2017

This paper details an application-specific integrated circuit (ASIC) with an array of switched-resistor-based memristors (resistor with memory) and integrate & fire (I & F) neuron circuits for the development of memristor-based pattern recognition. Since real memristors are not commercially available, a compact memristor emulator is needed for device study. The designed ASIC has five memristor emulators with one having a conductance range from 4.88 ns to 4.99 μs (200 kΩ to 204.8 MΩ) and other four having conductance ranging from 195 ns to 190 μs (5.2 kΩ to 5.12 MΩ). Signal processing has been planned to be off-chip to get the freedom of programmability of a wide range of memristive behavior. This paper introduces the memristor emulator and the realization of synapse functionalities used in neuromorphic circuits such as long term potentiation (LTP), Long Term depression (LTD) and synaptic plasticity. The ASIC has two I & F neuron circuits which are intended to be used in conjunction with memristors in a multiple chip network for pattern recognition. This paper explains the memristor emulator, I & F neuron circuit and a respective neuromorphic system for pattern recognition simulated in LTspice. The ASIC has been fabricated in AMS 350 nm process.

Keywords: ASIC; emulator; CMOS; LTD; LTP; memristor; neuron; synaptic plasticity; pattern recognition.

*This paper was recommended by Regional Editor Piero Malcovati.
§Corresponding author.

1. Introduction

The memristor was first postulated by Chua in 1971[1] as the fourth fundamental element like resistor, capacitor and inductor. He described the memristor (resistor with memory) as a two-terminal floating device which relates two physical quantities: flux and charge. Since then, lots of research have been done to find such a device with nonlinear voltage–current relation and memory of applied voltages. It was 2008, when a device was fabricated in the Hewlett–Packard (HP) lab which had the fundamental property of a memory–resistor, however, it was not based on the flux–charge relation. This nanoscale device was TiO_2-based, where the conductance changes with the applied voltage. It was realized by the transport of oxygen ions from doped to undoped region and vice-versa.[2] Since the development of memristor in 2008,[2] the memristor has drawn a lot of attention in the scientific community due to its forecasted application in nonvolatile memory,[3] programmable analog circuits,[4] neuromorphic circuits[5,6] and many more. The memristor behaves similar to the synapse of human brain, so it has opened a lot of possibilities in the design of neuromorphic circuits with these memristors and CMOS circuits. Due to the absence of commercially available real memristor hardware and the uncertainty about the memristor model to be used in neuromorphic circuit for better learning, a programmable memristor emulator is needed. Recently, Alharbi *et al.* [7] have simulated memristor emulator which can emulate single memristor behavior and has no programmable feature. Kumngern *et al.* [8] have developed emulator with commercial components which is nonprogrammable and it is quite big to be used in bigger network. Pershin *et al.*[4] have developed programmable memristor emulator with commercial components, but has low resistance range, low discretization step and high operating voltage. Keeping the requirements of neuromorphic circuit in mind, an array of memristor emulators has been designed which can be used within a bigger network for the proof of concept of biologically inspired memristive learning; at the same time, any memristor model can be implemented for analyzing its impact on learning.

The human brain has a dense neuron–synapse structure (10^{11} neurons and 10^{15} synapses) and the brain code is not fully understood yet. In order to understand the complex structure of the human brain, the first step is to understand the properties of synapses and their impact on learning. Using a programmable memristor emulator, the impact of synaptic plasticity and update rate under suitable pulse train on human learning can be studied with various synapse models.

2. Memristor Emulator Application-Specific Integrated Circuit (ASIC)

A chip containing an array of programmable memristor emulators and two integrate and fire (I & F) neuron circuits has been designed and fabricated in AMS 350 nm process. Figure 1 shows a photograph of the ASIC.

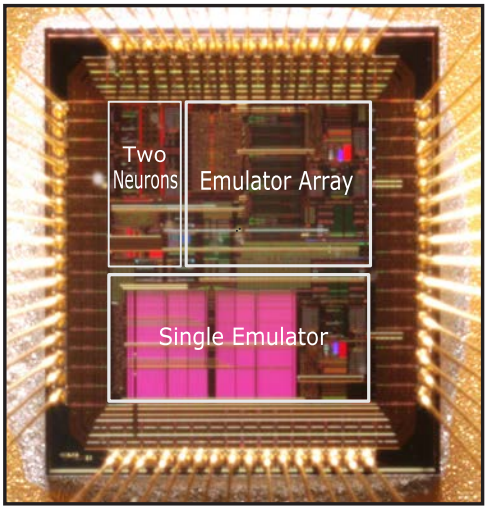


Fig. 1. Memristor emulator ASIC.

2.1. *Memristor model*

The memristor model implemented in this work has a state variable-dependent conductance which relates current and voltage described by

$$I(t) = G(x) \cdot v(t)\,, \tag{1}$$

where $I(t)$ is the current, $v(t)$ is the voltage, $G(x)$ is the conductance and x is the state variable. $G(x)$ is defined as

$$G(x) = G_{\min} + f_1(x) \cdot (G_{\max} - G_{\min})\,, \tag{2}$$

where $G_{\min}$ and $G_{\max}$ are the minimum and maximum conductances designed on chip, $f_1(x)$ is a function of state variable x, which can be programmed externally. The function $f_1(x)$ can be programmed to be a linear or nonlinear function of the state variable x which will determine the linear or nonlinear dependence of conductance on state variable x. In this work,

$$f_1(x) = x \tag{3}$$

was programmed, assuming a linear dependence of conductance on state variable.

The state variable x changes with the total amount of current flown in the past through the memristor and is modeled by

$$\frac{dx}{dt} = f(x, v) = k \cdot G(x) \cdot v(t) \cdot f_w(x)\,, \tag{4}$$

where $f_w(x)$ is a window function modeling the nonlinear dopant drift and limiting the state variable in the range of 0 to 1.[9] The window function used in this work is

proposed in Ref. 10 and described by

$$f_w(x) = 1 - (2x - 1)^2 + \delta\,. \tag{5}$$

A small parameter δ has been added in (5) in order to avoid the locking problem at the boundary.[9] δ can be considered as noise, the value chosen (0.0003) is very small compared to the maximum value of $f_w(x) = 1$.

In (4), k is the technology-dependent constant which has been taken from the model in Ref. 2 and is explained by

$$k = \frac{\mu}{D^2 \cdot G_{\max}}\,, \tag{6}$$

where μ is the dopant mobility, D is the device length and $G_{\max}$ is the maximum conductance of the device.

The dependence of the state variable on the history of the current flown through the memristor creates a memory effect for the device (memory–resistor) forming a pinched hysteresis loop between current and voltage.[11] Taking the parameters $\mu = 10^{-14}\,\mathrm{m^2 s^{-1} V^{-1}}$, $D = 10\,\mathrm{nm}$, $G_{\max} = 4.99\,\mu\mathrm{s}$ and $G_{\min} = 4.88\,\mathrm{ns}$, and using a Simulink model of the designed ASIC and the processing block in Matlab, current and voltage curves have been simulated. The simulation shows the characteristic pinched hysteresis curve as shown in Fig. 2. The curve has been drawn for the applied differential sinusoidal input voltage of 6 V_{pp} (peak to peak voltage) and initial

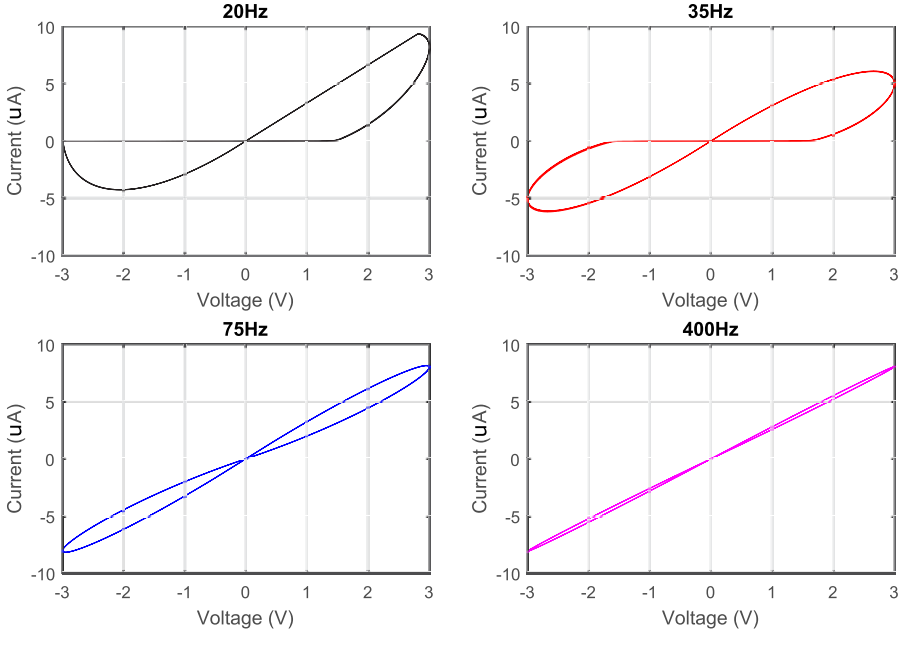


Fig. 2. Pinched hysteresis curve.

conductance (G_{initial}) of 3.41 μs for different frequencies. The input frequency has an impact on the hysteresis of memristor, because at lower frequencies, the ions can better follow the applied electric field, leading to more pronounced hysteresis at lower frequencies than at higher frequencies.

2.2. *Memristor emulator: Architecture and design*

The designed memristor emulator has on chip a resistor array and analog processing block, and an off-chip digital signal processing (DSP) unit (Fig. 3).

The voltage measurement and analog processing consist of a differential voltage measurement circuit, an analog to digital converter (ADC) and a parallel to serial converter. This block (Fig. 4) continuously measures the voltage and sends a 10-bit

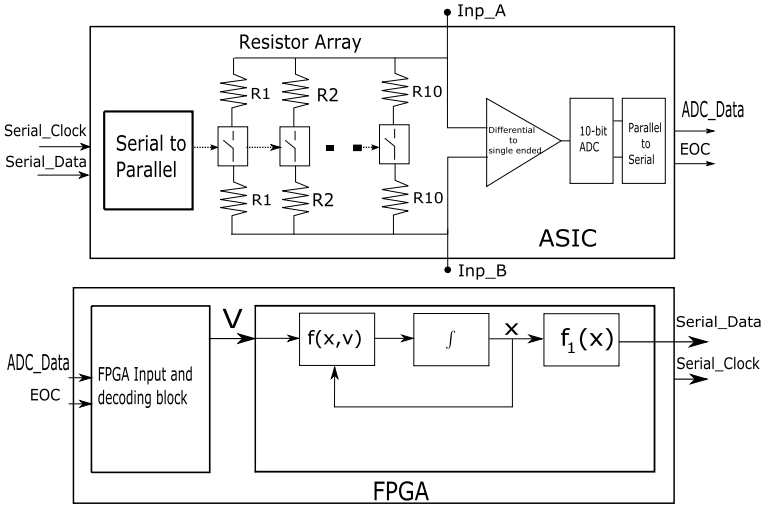


Fig. 3. Memristor emulator architecture.

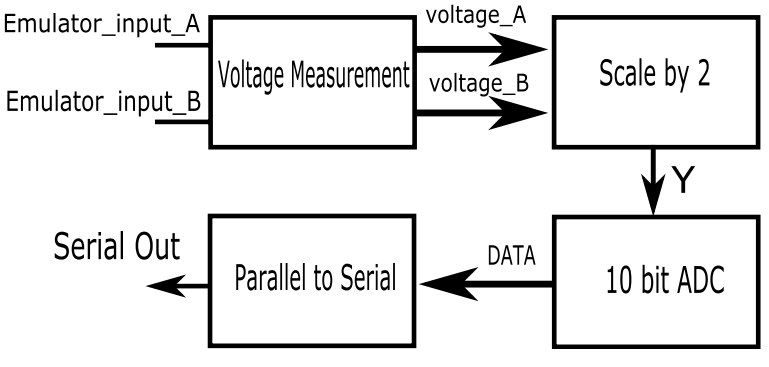


Fig. 4. Voltage measurement and analog processing.

Table 1. Comparison of a solid-state memristor and the present version of memristor emulator with other emulators.

Parameter	Real memristor	Memristor emulator	Ref. 4
Resistance range	Determined by the structure	$200\,\text{k}\Omega < R < 204.8\,\text{M}\Omega$	$50\,\Omega < R < 10\,\text{k}\Omega$
Discretization of R	R changes continuously	1024	256
Frequency	Any	$< 45\,\text{kHz}$	$< 50\,\text{Hz}$
Response	Determined by the structure	Determined by preprogrammed function	Determined by preprogrammed function
Applied V	Less than the breakdown voltage of structure	0, +3.3 V	0, +5 V or +2.5 V, −2.5 V
Supply V	Not needed	0, +3.3 V	0, +5 V or +2.5 V, −2.5 V

serial data to the FPGA shown in Fig. 3. The potential at the floating terminals (Emulator_input_A and Emulator_input_B) can vary between 0 V and 3.3 V, so the differential voltage across the terminals can vary between −3.3 V and 3.3 V. The ADC has the input voltage limits of 0 V and 3.3 V; in order to accommodate the full span of differential voltage, the measured signal has been scaled by a factor of 2 as shown in Fig. 4. The scaled signal is further converted into a 10-bit digital data by a 10-bit ADC. Parallel to serial shift register converts the parallel data from ADC into serial output format for the transfer to the FPGA.

The emulator circuit has a binary weighted resistor array between its floating terminals. The overall conductance between the terminals is digitally controlled through a series-connected switch to each resistor in the array as shown in Fig. 3. The serial to parallel block on-chip converts the serial data received from the FPGA into parallel (clock and inverted clock) data for the transmission gates between the resistors (Fig. 3).

Table 1 shows a summary of the main characteristics of a real memristor, present emulator and emulator in Ref. 4. In case of emulator, its characteristics are mainly limited by the area of chip, ADC and the processing time of the memristive function to be implemented.

2.3. *Data processing on FPGA*

The memristor emulator utilizes an FPGA for performing the computations required to emulate a specific model (Fig. 3). The model designed in the FPGA allows for periodically requesting the voltage at the ASIC inputs, computing the state variable and updating the ASIC conductance in less than one cycle of the serial clock. This processing unit is divided into two major modules: a communication module and a DSP module.

2.3.1. *Communication module*

The communication sub-modules were designed to work independently from the signal processing block, so that ADC reading, processing and state variable

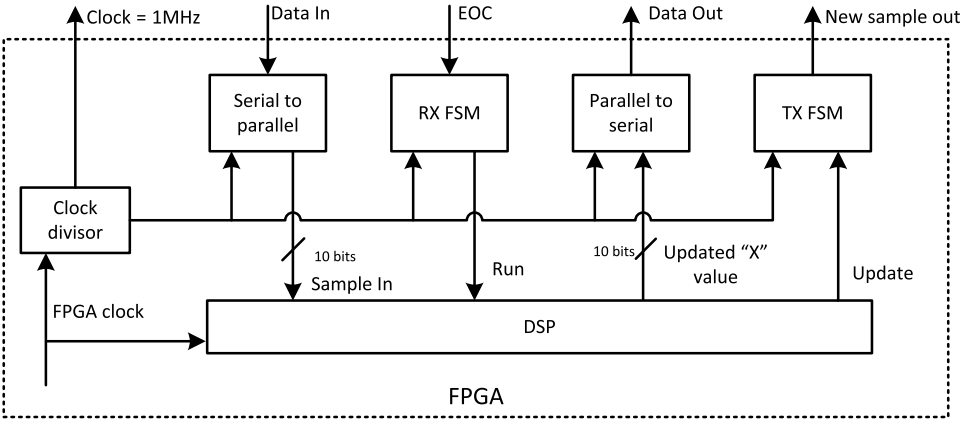


Fig. 5. Top block design in the FPGA.

transmission can be done in parallel. The serial clock is derived from the main FPGA clock and it is sent to the ASIC and is also used in the rest of the communication blocks for synchronizing the data as shown in Fig. 5. Two finite-state-machine (FSM) sub-blocks were designed in order to control the data acquisition and data transmission.

Both transmission and reception were designed by using FSMs. In the case of the data reception FSM, the initial state will be locked until the ASIC sends EOC signal. After that, the system samples the 10 bits on the falling edge of the clock and finally alerts the DSP block that a new voltage sample is available. In the case of the data transmission FSM, it is locked on the initial state until the DSP block triggers an update. At that point, a new 10-bit data value is present at the input of the parallel to serial block. The FSM sets the new sample pin to low and then it sends the bits at the falling edge of the clock.

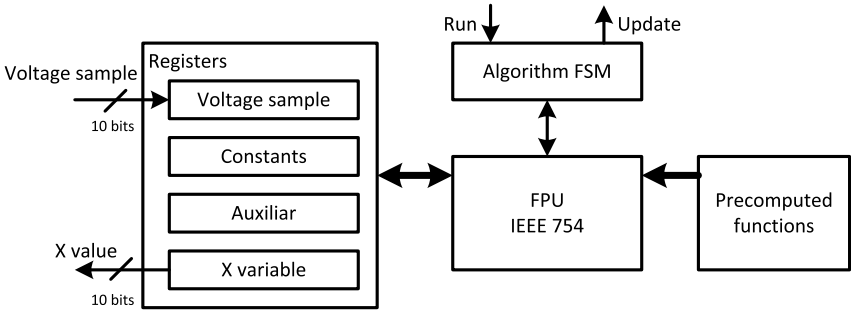


Fig. 6. DSP module.

2.3.2. *DSP unit*

The data processing is realized by using a custom made DSP unit as shown in Fig. 6. The designed processor has a floating-point unit which is able to compute addition, subtraction, multiplication, division and comparison of 32-bit numbers under the IEEE 754 standard. The unit is also able to compute the required conversions from integer numbers to floating point and vice versa.

A set of registers is used to store the incoming voltage sample and the updated value of the memristor state variable x. Apart from this, a sub-set of registers hold constant values (e.g., minimum and maximum conductance) and also intermediate results (e.g., current conductance value). In order to speed up the calculation time and to reduce complexity, the DSP unit has memory blocks dedicated to store precomputed functions (e.g., exponential function of the state variable, e^x) which are indexed by the integer value of the voltage sample or the integer value of the memristor state variable.

The algorithm is hardcoded on the DSP unit as an FSM and is shown in Fig. 7. After initialization, the FSM is ready to start a computation cycle, which will begin once it receives pulse in the Run input (triggered by the data reception module), then the DSP module reads the new voltage sample and computes the analog voltage value from its digital version. The voltage value is then used to compute $f(x, v)$ (4) and then to compute its discrete integral

$$Q^+ = Q^- + f(x, v) * Ts\,, \tag{7}$$

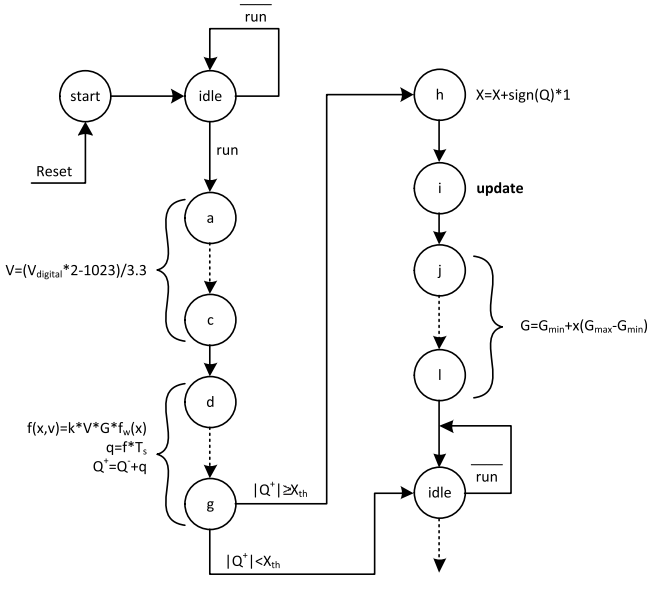


Fig. 7. Data processing unit state machine.

where Ts is the sampling period, Q^- is the present value of the variable Q (during initialization, this value is set to zero) and Q^+ is its updated value. In the next step, the value of Q is compared against the threshold value X_{th}. This threshold is the value of X that ensures a change of one bit on the discrete value of the memristor state variable X, and is defined by

$$X_{\text{th}} = \frac{D^2 G_{\max}}{\mu(2^n - 1)}\,, \tag{8}$$

where n is the word length of variable X. In this work, X has 10 bit of resolution ($n = 10$). If the magnitude of Q is above the threshold value, then the new value of X is computed and updated and the update flag is triggered, otherwise the state machine returns to its idle state without performing any update on the state variable. In the case of an update of the value of X, the system computes the new value of conductance (2) and this updated conductance value is stored and then used in subsequent computations until a new update on X is performed. At the end, the state machine moves to the idle state where it waits for a new cycle.

For these calculations, the values of $G_{\max}$, $G_{\min}$, k, Q_{th} and Ts are stored as constant values, and the function $f_w(x)$ is used as a precomputed table indexed by the integer value of X.

2.4. *Measurement results*

The memristor has a state-dependent resistance which can be measured by current and voltage across the emulator. The applied sine wave voltage across the memristor was traced on an oscilloscope together with the voltage across a fixed resistance R in series with the memristor terminals. R was chosen equal to 100 KΩ which is very small compared to the maximum resistance of the memristor (204.8 MΩ).

The voltage across the fixed resistor (R) was measured by using an isolation amplifier in cascade with a dc blocker, which removes the inherent offset introduced by the previous stage. The measuring setup is presented in Fig. 8. The voltage across

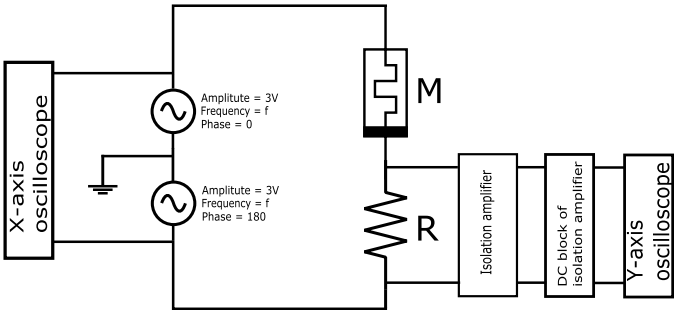
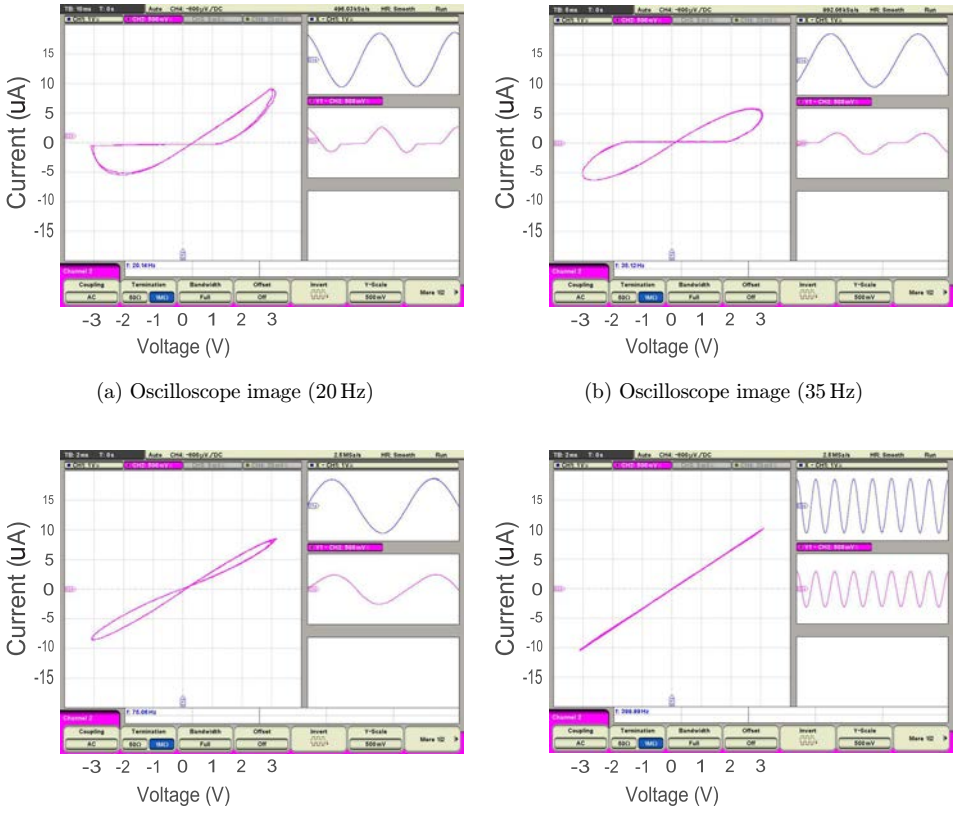


Fig. 8. Measurement setup.



(a) Oscilloscope image (20 Hz)

(b) Oscilloscope image (35 Hz)

(c) Oscilloscope image (75 Hz)

(d) Oscilloscope image (400 Hz)

Fig. 9. Oscilloscope screenshot of measurement on ASIC.

the fixed resistance is plotted in the Y-axis and the total applied voltage in the X-axis, giving as a result the oscilloscope screenshots shown in Fig. 9.

The memristor emulator ASIC has been tested with a differential sine wave signal of $+3$ V to -3 V and G_{initial} of 3.41 μs for the frequencies of 20, 35, 75, and 400 Hz. The pinched hysteresis curve measured on the emulator ASIC is shown in Fig. 9. The memristor behaves like a resistor at 400 Hz and it has a high hysteresis at a low frequency of 20 Hz. The measurement results show good agreement with the Matlab Simulink simulation as shown in Fig. 2. The corresponding time-domain waveform of voltage and current is displayed in Fig. 9 alongside the main window. The range of frequency depends on the model and constants used and can extend up to 45 kHz, which is limited by the sampling period of the ASIC of 11 μs where it takes 10 μs for sending new analog sample and 1 μs for signal processing in FPGA.

3. Long Term Potentiation (LTP) and Long Term Depression (LTD)

3.1. *LTP*

LTP[12,13] is a phenomenon in neuroscience in which synapses get strengthened (long lasting) with a burst of high frequency stimulation called tetanus in biophysical parlance. This phenomenon was first proved by the Bliss–Lomo experiment.[14] LTP is supposed to be responsible for storing information in our neural system. The memristor is supposed to be the replica of the synapses in neuromorphic engineering and it should show the LTP characteristic similar to real synapses in human brain.

The memristor emulator ASIC has been designed to be a part of bigger neural network for memristive learning, so LTP characteristic of memristor has been tested by applying a number of rectangular pulses of amplitude 3 V and a pulse width of 500 μs. The conductance is seen to increase after each pulse-like real synapses as shown in Fig. 10.

3.2. *LTD*

LTD is the inverse phenomena of LTP (Sec. 3.1) where the subsequent pulses in reverse polarity weaken the synapses and reduce the conductance towards the lowest value.[15] The LTD has been measured on the emulator where we see the decrease in the conductance across the memristor with the subsequent negative pulses (Fig. 10).

4. Synaptic Plasticity

Synaptic plasticity is the adjustment in the synaptic weight between two neurons based on the pulse width duration of applied spikes across the synapses. As explained

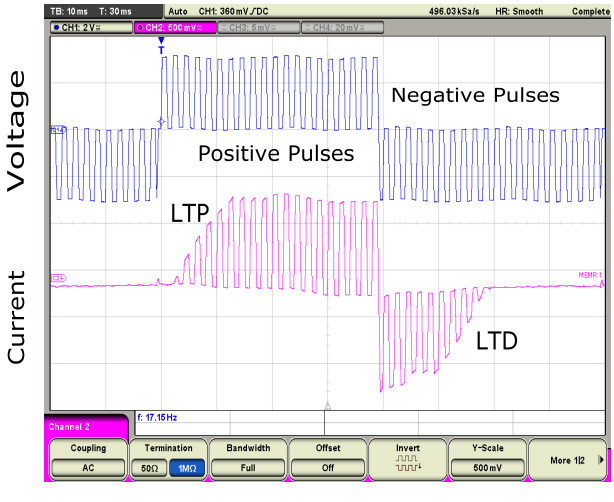


Fig. 10. LTP and LTD measurement.

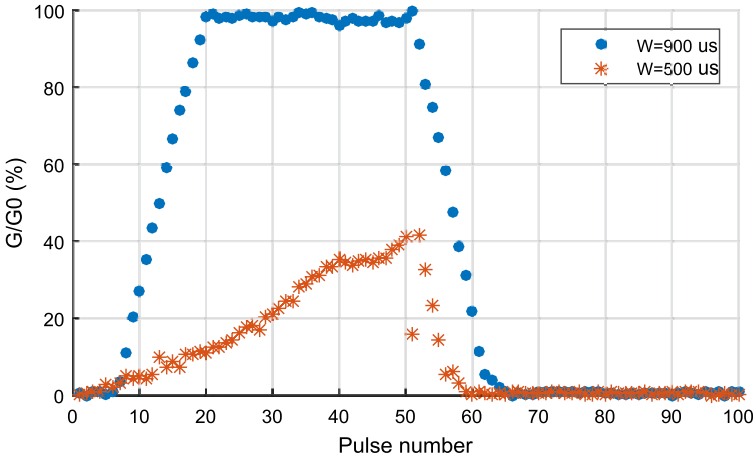


Fig. 11. Synaptic plasticity measurement.

in Ref. 6, synaptic plasticity based on pulse width is the core phenomena for Spike Time-Dependent Plasticity (STDP)[16,17] which is the basic mechanism for neural learning.[18] The memristor model used in this work has been tested with 50 potentiation and depression pulses of amplitude 1 V and pulse widths 500 μs and 900 μs in order to update the conductance at different rates. The measurement results show a higher rate of change of conductance for 900 μs compared to 500 μs for the same number of the applied pulses as shown in Fig. 11.

5. I & F Neuron

An analog I & F neuron circuit was designed on chip to emulate the principal behavior of biological neurons. It integrates all input currents and fires when a particular threshold voltage is reached.

I & F is widely used for implementing artificial neural networks with STDP.[5] STDP is the adjustment in synaptic weight between two neurons based on the relative input and output firing time. The schematic representation of the I & F neuron is shown in Fig. 12. The circuit operates in two different phases, integration and spiking, controlled by the digital input pin V_{ph}. The circuit receives and integrates multiple presynaptic current pulses at its analog input pin **in** till the integrated charge reaches a threshold and the neuron triggers an off-chip spike generator. The resting potential of input pin **in** is provided by the analog input V_{fb} during integration. The output transition (V_{out}) externally triggers a digital HIGH at V_{ph} in order to change the circuit operation from integration into spiking. Analogous to the refractory period of biological neurons, the circuit is unable to generate new spike events during this phase. In order to implement STDP, a spike will also be applied to pin V_{fb} during the spiking phase. The op-amp will mirror these spikes to the neuron

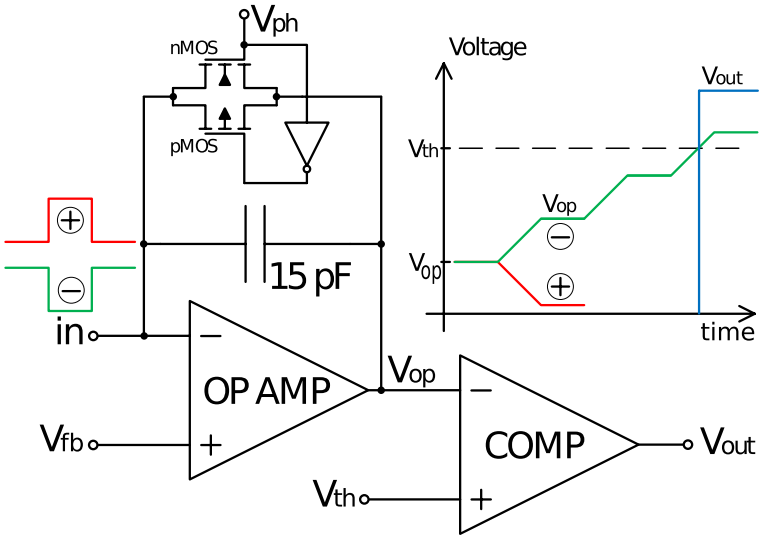


Fig. 12. I & F neuron (color online).

input that is connected to memristors. The circuit will stay in this phase for the whole duration of a spike. Once the spike is over, a digital LOW on V_{ph} will switch the circuit operation back to integration.

Excitatory and inhibitory synaptic effects are realized by changing the sign of the input pulses as illustrated in Fig. 12. The circuit configuration leads to excitatory effects by using negative input currents while inhibitory effects are realized by using positive currents. The circuit implementation creates the possibility to control the resting potential and spiking threshold externally, which gives the flexibility to build neuronal layers with diverse characteristics that can be assigned to various different computational tasks.

The response of the I & F neuron was tested by applying constant input currents and measuring the integration time needed until the circuit signals a fire event as shown in Fig. 13 for a specific set of input signals. The resting potential was set to 1 V and the threshold to 2.7 V while a negative current of 1 nA was applied at the input. In order to change between the two operation phases spiking and integration, a pulse train was applied to V_{ph}. A constant input current leads to a linear change of V_{op}. Once it crosses the threshold, the output pin makes a transition from LOW to HIGH, as illustrated in Fig. 12. The integration time can be measured on the oscilloscope display starting from the negative transition of V_{ph} (blue signal), which indicates the beginning of the integration, up to the moment V_{out} changes from LOW to HIGH (pink signal). The integration time can be calculated as

$$t = \frac{C \cdot (V_{\mathrm{th}} - V_{\mathrm{fb}})}{I_{\mathrm{in}}}\,, \tag{9}$$

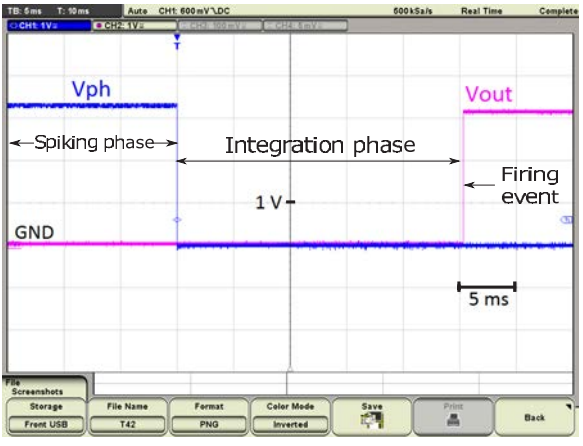


Fig. 13. I & F neuron measurement (color online).

where C is the feedback cap, V_{th} is threshold voltage, V_{fb} is resting potential and I_{in} is input current.

With the applied input signals and a 15 pF feedback capacitance, (9) yields an integration time of 25.5 ms. The integration time measured on the oscilloscope (Fig. 13) has been found to be 25.3 ms, which is in very good agreement with theory.

6. Memristive Pattern Recognition

Biologically inspired hardware that natively supports the basic functionality of neurons and synapses enables very efficient implementation of artificial neural networks (ANN). Memristors have been shown to mimic the behavior of synapses with respect to STDP. Since ANNs contain orders of magnitude of more synapses than neurons, their space efficient implementation is crucial. Memristors are orders of magnitude smaller than VLSI synapse implementations (e.g., Ref. 19). Suitable devices as described in Refs. 6 and 20–22, display exponential changes in memristor state when exposed to a voltage

$$\frac{dw}{dt} \propto \sinh(\beta V)\,. \tag{10}$$

A memristor connected to two neurons that fire simultaneously is exposed to higher across voltages and thus much greater state changes than for individual neuron pulses. This enables the realization of STDP and on a larger scale unsupervised pattern learning and recognition. For this work, a one-layer feed-forward ANN as described in Refs. 6 and 23 with an adapted input coding has been designed and simulated on circuit level in SPICE. In contrast to commonly used event-based simulators, no artificial threshold levels as in Ref. 5 could be used and the simulation

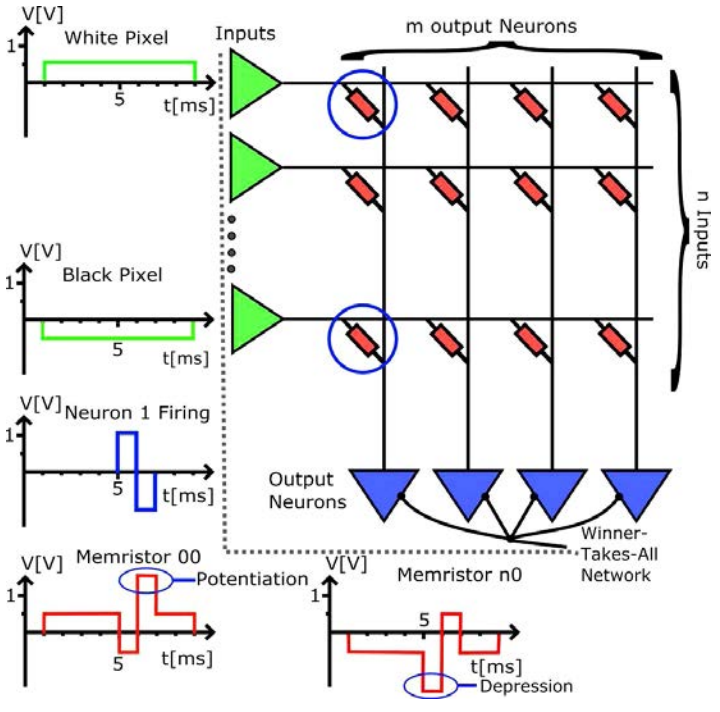


Fig. 14. Network architecture and mechanism.

paves the way towards a future hardware realization based on the memristor emulator and neuron explained in Sec. 2.1.

The memristors are arranged in a cross-bar array structure connected to inputs, which correspond to image pixels, and connected to output neurons as depicted in Fig. 14. Due to STDP, neurons that fire while an image is presented at the input adapt their receptive fields towards that image. Additional necessary elements of the network are competition, homeostasis and variability. Competition — the output neurons are connected in a winner-takes-all network. Thus, only the neuron that fires first can adapt its receptive field towards the current image. Homeostasis — this adapts the threshold voltage of the output neurons according to

$$\frac{dV_{\text{th}}}{dt} = \gamma(R_{\text{avg}} - R_{\text{target}}) \tag{11}$$

with the average firing rate R_{avg}, target firing rate R_{target} and the gain factor γ. This forces the individual output neurons to specialize on one pattern. Variability in the initial memristor states is the starting point for neuron differentiation. Finally, variability in the input coding is necessary for reliable learning of all the presented patterns. This has to be balanced with pattern recognizability.

The memristors were implemented based on a structure from Ref. 24 and optimized to replicate the experimental results in Ref. 6. The neurons were implemented according to the I & F architecture designed on the ASIC (Sec. 5). In addition, a behavioral spike generator was added whose output is fed back to the neuron input. Homeostasis was implemented using controlled voltage sources based on the differential equation (11). The gray-valued pixels are translated into SPICE stimulus voltages for the network inputs using a scheme based on Ref. 23. However, due to the lower number of pixels in realizable networks, the input coding had to be modified to more accurately represent pixel values using a dither-like technique (Algorithm 1). As an example, a network with 12 inputs and 4 output neurons connected via a receptive matrix of 48 memristors was simulated. The four patterns to be learned are shown in Fig. 15.

The development of the receptive fields of the output neurons during learning is shown in Fig. 16. The neurons quickly adapted their receptive fields, and after 40 cycles, two patterns have already been reproduced in two receptive fields. Initially, neurons 2 to 4 all converged to pattern number 3. However, due to competition, homeostasis and variability, neurons 2 and 4 reallocate their receptive fields towards the yet unrecognized patterns 1 and 4. After 200 cycles, the network had learned and was able to uniquely identify all four patterns. These simulations demonstrate that online

Algorithm 1. Input Coding

for Every 10 ms cycle **do**
  Randomly select one pattern
  Add gaussian noise to pixel values $\tilde{p}_i = p_i + g$
  for N subcycles **do**
    Generate Random Number $r_i \in [0, 1)$
    for All image pixels **do**
      if $|\tilde{p}_i| > r_i$ **then** Fire with $V = V_{\text{pulse}} \cdot \text{sgn}(\tilde{p}_i)$
      end if
    end for
  end for
end for

Fig. 15. Basic patterns to be learned by the ANN.

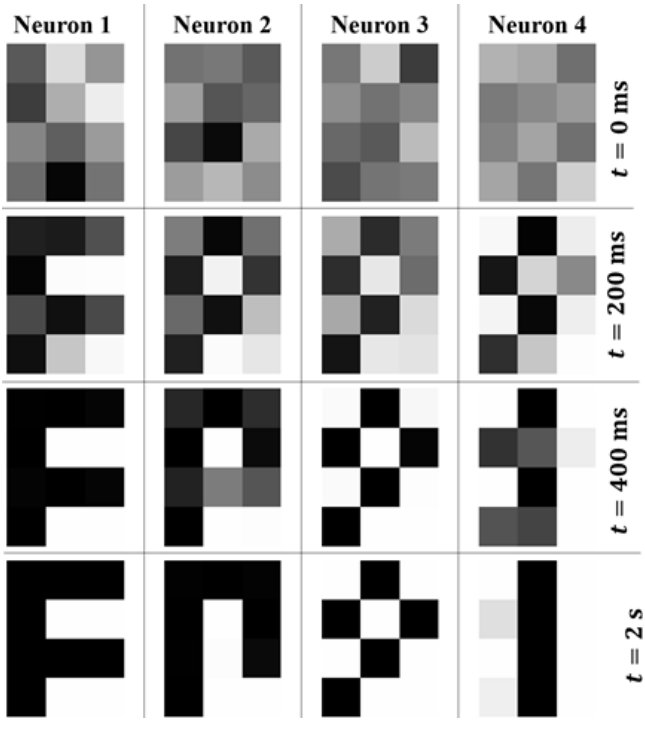


Fig. 16. Receptive development during learning phase. White pixels correspond to a memristor state of $w = 1$ and black to $w = 0$. The base patterns with added noise of 0.3 and SNR = 7.54.

pattern learning and recognition is possible on a hardware level with the memristive devices presented in Ref. 6 and the overall architecture described in Ref. 23. This architecture will be realized in the near future with the chip described in Sec. 2.

7. Conclusion and Future Work

Memristor Emulator ASIC with I & F neuron has been designed and the experimental tests have been carried out with success. A pinched hysteresis loop has been realized by programming the functions $f_1(x)$ and $f(x, v)$ externally in a FPGA which ensures that any function can be realized on the memristor emulator. The frequency of operation has an impact on the hysteresis of memristor. Tests have been performed for 20, 35, 75, and 400 Hz, which show that at higher frequencies the memristor behaves like a resistor and has higher hysteresis at lower frequencies. The memristor emulator shows the freedom of programmability and selectivity which can be used for realizing a bigger network for pattern recognition. The I & F neuron circuit shows the functionality similar to biological neuron and can be integrated with the array of memristors for neuromorphic circuit design.

A neural network circuit for memristive pattern recognition has been set up and simulated in LTspice with respective circuit models of memristors and neuron circuits. The simulations showed that the circuit was able to successfully learn and recognize four patterns in 3×4 pixel input images. In future, it is planned to realize the simulated pattern recognition in hardware using several of these chips.

Acknowledgments

Thanks to German Research Foundation (DFG) for funding the research group FOR2093 (Memristive Components for Neural Systems), Project C2 (Neural Circuits), under which this research activity has been conducted. A preliminary version of this paper has been presented at the 2016 39th International Conference on Telecommunication and Signal Processing (TSP) *Integrated Circuit with Memristor Emulator Array and Neuron Circuits for Neuromorphic Pattern Recognition*[25] *and Programmable Memristor Emulator ASIC for Biologically Inspired Memristive Learning.*[26]

References

1. L. O. Chua, Memristor-the missing circuit element, *IEEE Trans. Circuit Theory*, **18** (1971) 507–519.
2. D. B. Strukov, G. S. Snider, D. R. Stewart and R. S. Williams, The missing memristor found, *Nature*, **453** (2008) 80–83.
3. P. W. C. Ho, H. A. F. Almurib and T. N. Kumar, One-bit non-volatile memory cell using memristor and transmission gates, *2014 2nd Int. Conf. Electronic Design (ICED)*, 2014, pp. 244–248.
4. Y. V. Pershin and M. D. Ventra, Practical approach to programmable analog circuits with memristors, *IEEE Trans. Circuits Syst. I, Regular Papers*, **57** (2010) 1857–1864.
5. C. Zamarreño-Ramos, L. Camuñas-Mesa, J. A. Pérez-Carrasco, T. Masquelier, T. Serrano-Gotarredona and B. Linares-Barranco, On spike-timing-dependent-plasticity, memristive devices, and building a self-learning visual cortex, *Front. Neurosci.* **5** (2011) 00026.
6. F. Zahari, M. Hansen, T. Mussenbrock, M. Ziegler and H. Kohlstedt, Pattern recognition with tio x-based memristive devices, *AIMS Mater. Sci.* **2** (2015) 2372–0484.
7. A. G. Alharbi, M. E. Fouda and M. H. Chowdhury, Memristor emulator based on practical current controlled model, *2015 IEEE 58th Int. Midwest Symp. Circuits and Systems (MWSCAS)*, 2015, pp. 1–4.
8. M. Kumngern, A floating memristor emulator circuit using operational transconductance amplifiers, *2015 IEEE Int. Conf. Electron Devices and Solid-State Circuits (EDSSC)*, 2015, pp. 679–682.
9. Z. Biolek, D. Biolek and V. Biolkova, SPICE model of memristor with nonlinear dopant drift, *Radioengineering*, **18** (2009) 210–214.
10. Y. N. Joglekar and S. J. Wolf, The elusive memristor: Properties of basic electrical circuits, *Eur. J. Phy.* **2** (2009) 1–24.
11. S. Shin, L. Zheng, G. Weickhardt, S. Cho and S.-M. Kang, Compact circuit model and hardware emulation for floating memristor devices, *IEEE Circuits Syst. Mag.* **13** (2013) 42–55.

12. T. V. P. Bliss *et al.*, A synaptic model of memory: Long-term potentiation in the hippocampus, *Nature*, **361** (1993) 31–39.
13. G. M. Durand, Y. Kovalchuk and A. Konnerth, Long-term potentiation and functional synapse induction in developing hippocampus, *Nature*, **381** (1996) 71–75.
14. T. V. Bliss and T. Lømo, Long-lasting potentiation of synaptic transmission in the dentate area of the anaesthetized rabbit following stimulation of the perforant path, *J. Physiol.* **232** (1973) 331–356.
15. M. Ito, Long-term depression, *Annu. Rev. Neurosci.* **12** (1989) 85–102.
16. B. Linares-Barranco and T. Serrano-Gotarredona, Memristance can explain spike-time-dependent-plasticity in neural synapses, *Nat. Proc.* **1** (2009) 2009.
17. Y. Dan and M.-M. Poo, Spike timing-dependent plasticity: From synapse to perception, *Physiol. Rev.* **86** (2006) 1033–1048.
18. N. Caporale and Y. Dan, Spike timing-dependent plasticity: A hebbian learning rule, *Annu. Rev. Neurosci.* **31** (2008) 25–46.
19. J. Schemmel, A. Grübl, K. Meier and E. Mueller, Implementing synaptic plasticity in a vlsi spiking neural network model, *The 2006 Int. Joint Conf. Neural Networks IJCNN'06*, 2006, pp. 1–6.
20. T. Chang, S.-H. Jo, K.-H. Kim, P. Sheridan, S. Gaba and W. Lu, Synaptic behaviors and modeling of a metal oxide memristive device, *Appl. Phys. A* **102** (2011) 857–863.
21. S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder and W. Lu, Nanoscale memristor device as synapse in neuromorphic systems, *Nano Lett.* **10** (2010) 1297–1301.
22. M. D. Pickett, D. B. Strukov, J. L. Borghetti, J. J. Yang, G. S. Snider, D. R. Stewart and R. S. Williams, Switching dynamics in titanium dioxide memristive devices, *J. Appl. Phys.* **106** (2009) 074508.
23. D. Querlioz, O. Bichler and C. Gamrat, Simulation of a memristor-based spiking neural network immune to device variations, *The 2011 Int. Joint Conf. Neural Networks (IJCNN)*, 2011, pp. 1775–1781.
24. Y. V. Pershin and M. Di Ventra, SPICE model of memristive devices with threshold, *Radioengineering*, **22** (2013) 485–489.
25. R. Ranjan, A. Kyrmanidis, W. L. Hellweg, P. M. Ponce, L. A. Saleh, D. Schroeder and W. H. Krautschneider, Integrated circuit with memristor emulator array and neuron circuits for neuromorphic pattern recognition, *39th Int. Conf. Telecommunications and Signal Processing (TSP 2016)*, Vienna, Austria, 27–29 June 2016, pp. 265–268.
26. R. Ranjan, P. M. Ponce, A. Kankuppe, B. John, L. A. Saleh, D. Schroeder and W. H. Krautschneider, Programmable memristor emulator asic for biologically inspired memristive learning, *39th Int. Conf. Telecommunications and Signal Processing (TSP 2016)*, Vienna, Austria, 27–29 June 2016, pp. 261–264.