Valeri MLADENOV, Stoyan KIRILOV

Technical University of Sofia, Republic of Bulgaria

doi:10.15199/48.2016.04.06

Synthesis and Analysis of a Memristor-Based Perceptron for Logical Function Emulation

***Abstract**. The purpose of this research is to propose a new memristor-based synaptic device for use in perceptrons. A synaptic circuit made by two memristors is analyzed and a linear relationship between time and synaptic weight is obtained for rectangular input pulses. For adjusting the synaptic weight pulses with long duration and high magnitude are used. The operating input signals are with short duration and low amplitude to avoid altering the memristor state. A successful operation of the new memristor linear synapse is established after scaling the synaptic weight.*

***Streszczenie.** W pracy zaproponowano nowe urządzenie synaptyczne bazujące na memristorze, które można użyć w perceptronach. Przeanalizowano obwód synaptyczny zbudowany z dwóch memristorów i dla prostokątnych impulsów wejściowych uzyskano liniowe zależności pomiędzy czasem a wagą synaptyczną. W celu dopasowania wag synaptycznych użyto impulsów o długim czasie trwania i dużej amplitudzie. Sygnały wejściowe posiadają krótki czas trwania oraz małą amplitudę w celu uniknięcia zmiany stanu memristora. Po wyskalowaniu wagi synaptycznej uzyskano skuteczne działanie nowego memristora. (**Synteza i analiza perceptronu opartego na memristorze dla emulacji funkcji logicznej**)*

Keywords: memristor, synaptic weight, perceptron, activation function.
Słowa kluczowe: memristor, waga synaptyczna, perceptron, funkcja aktywacyjna.

Introduction

The artificial neural network simulates the structure, the processing methods and the system operation of biological neural system [1]. Many researches are attracted by neural networks advantages and applications – good performance, parallel processing, self learning, self adapting and fault tolerance. The perceptron is simplest kind of neural network, it is a nonlinear neuron and it can implement basic learning and parallel processing and also it can emulate simple logical functions [2, 3]. For the classical neurons CMOS weighting circuits and software implementations are used in synaptic circuits [4]. In the last few years, many scientists pay attention to memristor-based neural networks because memristor elements and circuits are capable of emulating the biological synapses by changing and retention of their state [5, 6]. The memristor element was predicted in 1971 by Prof. Leon Chua and its physical prototype was created in 2008 by Stanley Williams in the Hewlett-Packard research labs [7, 8]. The basic useful property of this new nonlinear electrical element is memorizing the amount of charge that has passed through it. The memristor is non-volatile element and it could retain its state and resistance for a very long interval. This ability is useful for application of the memristor in neural synapses schematics. For simulating new memristor-based neural networks several different models were used in the researches [8, 9, 10]. Also several neural memristor synapses schematics are investigated, like bridge-synapse and simple one-memristor synapse for neurons [4, 13]. One of the main advantages of the memristor-based synapses is the nanoscale dimensions of memristor and the ability for very-high dense integration of memristor crossbar-like structures [11, 15, 16, 17, 18]. The lack of analysis of linear memristor-based synapses with two serial-biased memristors is a prerequisite for the next investigations [11, 12, 13, 14, 15, 18].

In Section 2 a perceptron circuit for emulating the logical functions *OR* and *AND* is analyzed. The memristor-based serial synapse model is analyzed and validated in Section 3. In Section 4 the concluding remarks are presented.

A Simple Perceptron Circuit Investigation

The perceptron for emulating the logical functions *OR* and *AND* is presented in Fig. 1 [2]. The current time steps number is denoted with *t*. The input binary signals $x_1(t)$ and $x_2(t)$ are multiplied by the initial synaptic weights $w_1(t)$ and $w_2(t)$. The weight coefficients are based on the transfer function of a circuit containing memory elements – memristors, and a possibility for updating the weights exists with changing the memristor elements state variables. Then the signals obtained are submitted to a summing device with a shift coefficient b and the output signal $s(t)$ of the summing device is given with the next formula [2, 3]:

(1) $$s(t) = x_1(t)w_1(t) + x_2(t)w_2(t) + b$$

The signal $s(t)$ then is set to a device with relay activation function and with an activation threshold of $\Theta = 0$. The signal $y(t)$ after the relay element output is [2, 3]:

(2) $$y(t) = stp[s(t)] = \begin{cases} 1, & s(t) \geq \Theta \\ 0, & s(t) \prec \Theta \end{cases}$$

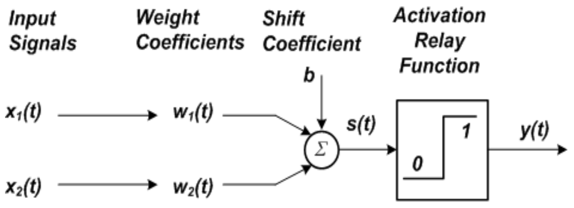

Fig.1. Schematic of a simple classical two-input perceptron circuit

The error signal $e(t)$ is obtained using the current value of the output signal $y(t)$ and the desired output signal $d(t)$ evaluated with respect to the logical functions *OR* and *AND* respectively [2, 3]. The difference between the signal $d(t)$ and the output signal $y(t)$ is the error signal $e(t)$ [2, 3]:

(3) $$e(t) = d(t) - y(t)$$

The error signal $e(t)$ is then multiplied by the input signals $x_1(t)$ and $x_2(t)$, respectively and after that we obtain the correcting adjustments for the synaptic weights w_1 and w_2 and for the shift coefficient b also [1, 2]:

(4) $$\begin{aligned} \Delta w_1(t) &= e(t)x_1(t) \\ \Delta w_2(t) &= e(t)x_2(t) \\ \Delta b(t) &= e(t) \end{aligned}$$

The new values of the synaptic weights and the shift coefficient are obtained by summing their old values with the synaptic weights adjustments:

$$w_{1new}(t+1) = w_1(t) + \Delta w_1(t)$$
$$w_{2new}(t+1) = w_2(t) + \Delta w_2(t) \qquad (5)$$
$$b_{new} = b_{old}(t) + \Delta b(t)$$

The initial values of the synaptic weights and of the shift coefficient are chosen with respect to the possibility for separating the output values of the logical 1 and 0 for the logical functions *OR* and *AND*. The initial data are set to the inputs of the perceptron scheme. The results from the training algorithm of the neural network are presented in Table 1 and Table 2, respectively. The results confirm that this perceptron successfully emulates the logical functions *OR* and *AND* for training and learning the neural network for 4 epochs. The 5-th epoch is additional and it is put to confirm that after adjusting the synaptic weights they remain constants if we apply another logical sequence of the binary input signals x_1 and x_2. After the final epoch the error signal is $e(t) = 0$. The weights adjustments Δw and Δb have values of -1 and 1. Using the final values of the synaptic weights of adjusted perceptron from Table 1 and Table 2 and formulas (1) and (2) we can obtain the dividing line equation for the logical functions *OR* and *AND* respectively [1, 2]:

$$w_{1final}x_1 + w_{2final}x_2 + b_{final} - \Theta = 0 \qquad (6)$$

Table 1. Results from the Perceptron Learning Process For Emulating the Logical Function *OR*

Epoch	x1	x2	d	w1	w2	b	x1*w1	x2*w2	s	y	e	Δw1	Δw2	Δb	w1_new	w2_new	b_new
1	0	0	0	0	0	0	0	0	0	1	-1	0	0	-1	0	0	-1
	0	1	1	0	0	-1	0	0	-1	0	1	0	1	1	0	1	0
	1	0	1	0	1	0	0	0	0	1	0	0	0	0	0	1	0
	1	1	1	0	1	0	0	1	1	1	0	0	0	0	0	1	0
2	0	0	0	0	1	0	0	0	0	1	-1	0	0	-1	0	1	-1
	0	1	1	0	1	-1	0	1	0	1	0	0	0	0	0	1	-1
	1	0	1	0	1	-1	0	0	-1	0	1	1	0	1	1	1	0
	1	1	1	1	1	0	1	1	2	1	0	0	0	0	1	1	0
3	0	0	0	1	1	0	0	0	0	1	-1	0	0	-1	1	1	-1
	0	1	1	1	1	-1	0	1	0	1	0	0	0	0	1	1	-1
	1	0	1	1	1	-1	1	0	0	1	0	0	0	0	1	1	-1
	1	1	1	1	1	-1	1	1	1	1	0	0	0	0	1	1	-1
4	0	0	0	1	1	-1	0	0	-1	0	0	0	0	0	1	1	-1
	0	1	1	1	1	-1	0	1	0	1	0	0	0	0	1	1	-1
	1	0	1	1	1	-1	1	0	0	1	0	0	0	0	1	1	-1
	1	1	1	1	1	-1	1	1	1	1	0	0	0	0	1	1	-1
5	0	0	0	1	1	-1	0	0	-1	0	0	0	0	0	1	1	-1
	0	1	1	1	1	-1	0	1	0	1	0	0	0	0	1	1	-1
	1	0	1	1	1	-1	1	0	0	1	0	0	0	0	1	1	-1
	1	1	1	1	1	-1	1	1	1	1	0	0	0	0	1	1	-1

Table 2. Results from the Perceptron Learning Process For Emulating the Logical Function *AND*

Epoch	x1	x2	d	w1	w2	b	x1*w1	x2*w2	s	y	e	Δw1	Δw2	Δb	w1_new	w2_new	b_new
1	0	0	0	0	0	0	0	0	0	1	-1	0	0	-1	0	0	-1
	0	1	0	0	0	-1	0	0	-1	0	0	0	0	0	0	0	-1
	1	0	0	0	0	-1	0	0	-1	0	0	0	0	0	0	0	-1
	1	1	1	0	0	-1	0	0	-1	0	1	1	1	1	1	1	0
2	0	0	0	1	1	0	0	0	0	1	-1	0	0	-1	1	1	-1
	0	1	0	1	1	-1	0	1	0	1	-1	0	-1	-1	1	0	-2
	1	0	0	1	0	-2	1	0	-1	0	0	0	0	0	1	0	-2
	1	1	1	1	0	-2	1	0	-1	0	1	1	1	1	2	1	-1
3	0	0	0	2	1	-1	0	0	-1	0	0	0	0	0	2	1	-1
	0	1	0	2	1	-1	0	1	0	1	-1	0	1	-1	2	2	-2
	1	0	0	2	2	-2	2	0	0	1	-1	-1	0	-1	1	2	-3
	1	1	1	1	2	-3	1	2	0	1	0	0	0	0	1	2	-3
4	0	0	0	1	2	-3	0	0	-3	0	0	0	0	0	1	2	-3
	0	1	0	1	2	-3	0	2	-1	0	0	0	0	0	1	2	-3
	1	0	0	1	2	-3	1	0	-2	0	0	0	0	0	1	2	-3
	1	1	1	1	2	-3	1	2	0	1	0	0	0	0	1	2	-3
5	0	0	0	1	2	-3	0	0	-3	0	0	0	0	0	1	2	-3
	0	1	0	1	2	-3	0	2	-1	0	0	0	0	0	1	2	-3
	1	0	0	1	2	-3	1	0	-2	0	0	0	0	0	1	2	-3
	1	1	1	1	2	-3	1	2	0	1	0	0	0	0	1	2	-3

The classification results for the adjusted memristor perceptron are presented in graphical form in Fig. 2 and Fig. 3 respectively.

A Simple Memristor-Based Synaptic Circuit Analysis and Calculations

The schematic of the new memristor-based synapse device is presented in Fig. 4. It consists of two serial-connected memristors in opposite directions.

For simplifying the circuit analysis the linear drift memristor model [8] and the limiting conditions of the Boundary Condition Memristor Model (BCM) [9] are used. The basic differential equation for the memristor elements used in the circuit is [8, 9]:

$$\frac{dx}{dt} = \eta \frac{\mu R_{ON}}{D^2} i \qquad (7)$$

where x is the state variable, η is a polarity coefficient with values of 1 or -1, μ = 1.10^{-12} $m^2/(V.s)$ is the ionic drift mobility, R_{ON} = 100 Ω is the resistance of the memristor in a fully closed state, D = 10 nm is the length of the memristor element, and i is the intensity of the current flowing through the memristor circuit.

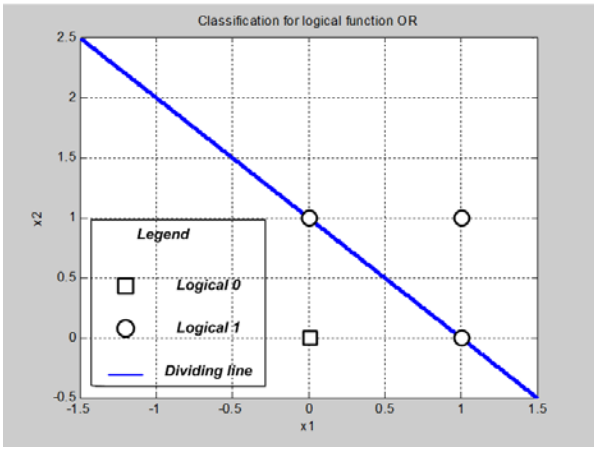

Fig. 2. The classification result of the adjusted perceptron for emulation the logical function *OR* according to Table 1

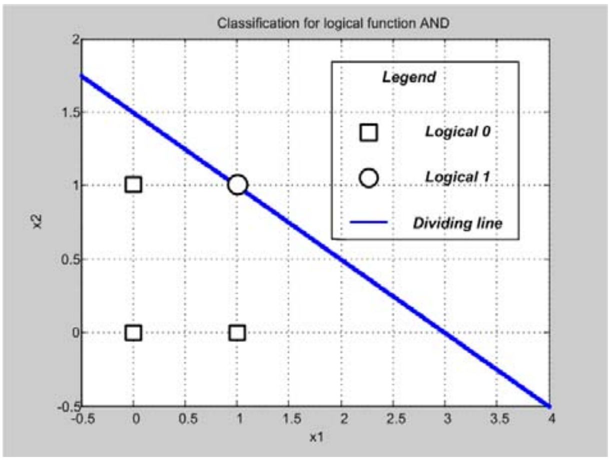

Fig. 3. The classification result of the adjusted perceptron for emulation the logical function *AND* according to Table 2

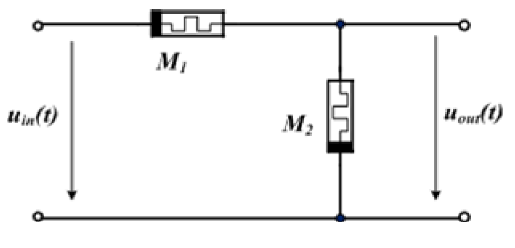

Fig. 4. Memristor-based serial synapse circuit

After integrating (7) we obtain the analytical expression of the state-flux relationship of the memristor element [8]:

$$x = x_0 + \eta \frac{\mu R_{ON}}{D^2} \int_{t_0}^{t} i dt' \tag{8}$$

The quantity x_0 is the initial value of state variable of the memristor [8]. The state variables for the memristors M_1 and M_2 will be denoted with $x^{|}$ and $x^{||}$ respectively. For the circuit investigated in initial moment the first memristor M_1 is set in fully open state and $x^{|}_0 = 0$, and the polarity coefficient is $\eta = 1$ because the memristor is forward-biased. The second memristor is set in a fully closed state and the state variable is $x^{||}_0 = 1$. Due to its reverse-biasing in the circuit with respect to the first memristor the coefficient η has a value of -1. The memristance of the first memristor M_1 is expressed with the next formula [8]:

$$M_1\left(x'\right) \approx R_{OFF}\left(1 - x'\right) = R_{OFF}\left(1 - \frac{\mu R_{ON}}{D^2} \int_{t_0}^{t} i dt'\right) \tag{9}$$

The resistance of the second element M_2 is [8]:

$$M_2\left(x''\right) \approx R_{OFF}\left(1 - x''\right) = R_{OFF} \frac{\mu R_{ON}}{D^2} \int_{t_0}^{t} i dt' \tag{10}$$

The quantity R_{OFF} presents the resistance of the memristor element in a fully open state. According to the Kirchhoff's Voltage Law and using (9) and (10) we obtain the equivalent resistance of the memristor-based synaptic circuit R_{eq} in a given moment t:

$$R_{eq}\left(t, x', x''\right) = M_1\left(x'\right) + M_2\left(x''\right) = R_{OFF} \tag{11}$$

The transfer function (synaptic weight) w of the memristor-based synapse given in Fig. 4 is:

$$w(t) = \frac{M_2\left(x''\right)}{M_1\left(x'\right) + M_2\left(x''\right)} = \frac{\mu R_{ON}}{D^2} \int_{t_0}^{t} i dt' = k \int_{t_0}^{t} i dt' \tag{12}$$

Using the Ohm's Law for the same circuit we can obtain a similar relationship but using a voltage input signal u_{in}:

$$w(t) = k \int_{t_0}^{t} i dt' = k \int_{t_0}^{t} \frac{u_{in}}{R_{OFF}} dt' = \frac{k}{R_{OFF}} \int_{t_0}^{t} u_{in}(t) dt' \tag{13}$$

Formulae (12) and (13) give us the possibility for adjusting the synaptic weight to use rectangular current or voltage pulses with an appropriate duration [4]. For positive weight adjustment a positive pulses are needed and after using negative pulse voltage we will have negative weight adjustment of the synaptic weight. In the initial moment the synaptic weight of the memristor-based synaptic circuit has its minimal value w_{min} [7, 8, 9]:

$$w_{\min} = \frac{M_2\left(x''_{\min}\right)}{M_1\left(x'_{\max}\right) + M_2\left(x''_{\min}\right)} = \tag{14}$$

$$= \frac{100}{16000 + 100} \approx 0.0062$$

The maximal synaptic weight w_{max} for the memristor-based circuit is [7, 8, 9]:

$$w_{\max} = \frac{M_2\left(x''_{\max}\right)}{M_1\left(x'_{\min}\right) + M_2\left(x''_{\max}\right)} = \frac{16000}{100 + 16000} \approx 0.9937 \tag{15}$$

After comparing the range of adjustment of the memristor synaptic weight with the results presented in Table 1 and Table 2, it is clear that the weight range of the circuit proposed could not covers the adjustment range needed for the logical functions emulation. But if we use a scaling mathematical process we can expand and translate the first range so that it will coincide with the range needed. The length of the first range is Δw_{1max} = 0.9937 – 0.0062 = 0.9875. The second range needed has the following length: Δw_{2max} = 2 – (-3) = 5. The ratio of $a = \Delta w_{2max} / \Delta w_{1max}$ = 5.0633 will be used for amplification the output signal of the memristor circuit which is the synaptic weight value for input pulse voltage with a magnitude of 1. After the amplification a summing process of the signal obtained with a constant voltage with a value of u_1 = -3 – (0.0062*5.0633) = -3.0314 is used to make match the synaptic weight intervals.

According to Tables 1 and 2 the synaptic weight adjustment has values of Δw = 1 or Δw = -1. Using division with the ratio a = 5.0633 we obtain the real synaptic adjustment for the memristor circuit: $|\Delta w_{real}|$= 1/5.0633 = 0.1975. If a rectangular pulse with a magnitude of 1 V is applied to the circuit input and using (13) we can calculate the pulse duration needed for adjusting the synaptic weight:

$$\Delta t_i = \frac{\Delta w_{real} R_{OFF}}{k u_m} = \frac{0.1975 \cdot 16000}{1.10^6.1} = 0.0032\ s \tag{15}$$

The same input of the synaptic circuit is used for applying the operating pulses needed for coding the signals x_1 and x_2. To avoid altering the memristors states during the operating signals the duration of the logical signals x_1 and x_2 have to be many times shorter with respect to the pulse width of the adjusting signals. Also we can choose the logical signals level to be several times lower than the adjusting signals.

To avoid shifting of the boundary between the doped and un-doped regions of the memristors due to the memorizing effect we could use the following levels for the input signals x_1 and x_2: for logical 1 a voltage pulse with a level of 0.2 V and for logical 0 – a level of - 0.2 V, respectively. The informational signals x_1 and x_2 could have duration with a value of 320 μs – about 10 times shorter than the width of the adjusting signals.

For applying the informational and adjusting signals to the memristor-based synapse input we could use different time intervals and also a multiplexer device is needed. A similar perceptron for emulating the logical function OR could be realized in MATLAB environment using the Neural Network Toolbox [14]. Follows the MATLAB code for the simulation:

```
x = [0 0 1 1; 0 1 0 1];
t = [0 1 1 1];
net = perceptron;
net = train(net,x,t);
view(net)
y = net(x);
```

The vector x presents input sequences of the binary signals x_1 and x_2. The array t presents the desired output signal d of the logical function OR. After construction and training the perceptron circuit, with the functions *perceptron* and *train*, respectively, the desired output signal y is obtained after 3 epochs.

The structure scheme of the software perceptron realized in MATLAB environment is obtained after applying the command *view* and the scheme is presented in Fig. 5.

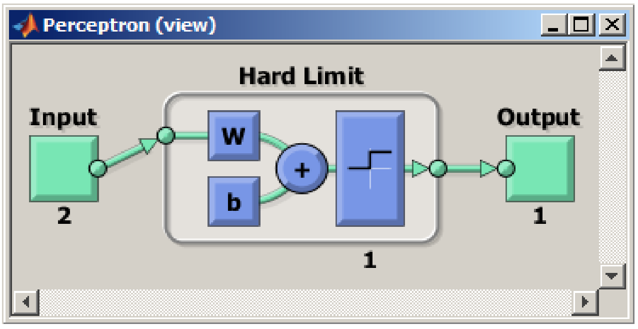

Fig. 5. Structure of the software-based perceptron circuit

Conclusions

The neural networks have very important applications in the science – especially for pattern recognition, function approximations, logical functions emulations, linear separation of patterns and many others. In the present investigation a new serial memristor-based synaptic circuit is presented and analyzed using the linear dopant drift memristor model and the limiting conditions of the BCM model also.

It is important that the inner resistance of the synapse remains constant during the training process. The transfer function of the memristor synapse is a linear function of time for rectangular voltage pulse sequences.

For the operating binary logical input signals short bipolar voltage pulses with low level have to be applied – positive for logical 1 and negative for logical 0. The adjusting signal has to be a sequence of rectangular voltage pulses with high level and long pulse duration so to be able to change the memristor state and conductance respectively.

The adjusting signals and the informational binary logical signals have to be applied in the same input of the memristor synapse but using different time intervals, according to the learning rule described with the formulas presented above. The classification results are presented and the possibility for separating the logical signals for 0 and 1 is confirmed.

For confirmation the result obtained a perceptron for emulating the logical function *OR* is realized in MATLAB environment also. The learning process is finished for shorter time interval.

The memristor-based neural networks have several advantages – nanoscale dimensions and high density of integration in chips, long term state retention and a good possibility for integration the CMOS and memristor layers in a single chip.

Power consumed for memristor chips is several times lower than the power consumed by classical neural integrated circuits and due to this the efficiency of the new memristor-based chips is higher. The memristor synapses are non-volatile unlike classical CMOS neural networks which require refreshing the synaptic weights.

Authors: *Prof. D-r Eng. Valeri Mladenov, Assist. Prof. D-r Eng. Stoyan Kirilov, Technical University of Sofia, 1000 Sofia 8 St. Kl. Ohridski Blvd, Dept. of Theoretical Electrical Engineering, E-mails: valerim@tu-sofia.bg; s_kirilov@tu-sofia.bg.*

REFERENCES

[1] Wang, L., M. Duan, S. Duan. Memristive Perceptron for Combinational Logic Classification. *Hindawi Publishing Corporation, Mathematical Problems in Engineering*, Vol. 2013, Article ID 625790, pp. 1 – 7, DOI:10.1155/2013/625790.
[2] Fausett, L. Fundamentals of Neural Networks. *Prentice Hall*, 1994, ISBN 0130422509.
[3] Haykin, S. Neural Networks: A Comprehensive Foundation. 2-nd edition, *Prentice Hall*, 1999, ISBN 0132733501.
[4] Sah, M., C. Yang, H. Kim, T. Roska, L. Chua. Memristor Bridge Circuit for Neural Synaptic Weighting. *13th International Workshop on Cellular Nanoscale Networks and Their Applications (CNNA), IEEE*, pp. 1 – 5, DOI: 10.1109/CNNA.2012.6331434, 2012.
[5] Jo, S., T. Chang, I. Ebong, B. Bhadviya, P. Mazumder, W. Lu. Nanoscale Memristor Device as Synapse in Neuromorphic Systems. *NanoLetters, American Chemical Society*, 2010, pp. 1297 – 1301, DOI: 10.1021/nl904092h.
[6] Kim, Y., K. Min. Synaptic Weighting Circuits for Cellular Neural Networks. *13th International Workshop on Cellular Nanoscale Networks and Their Applications (CNNA*), IEEE, pp. 1 – 6, DOI: 10.1109/CNNA.2012.6331430.
[7] Chua, L. O. Memristor – The Missing Circuit Element. *IEEE Trans. on Circuit Theory*, Vol. CT-18, pp. 507-519, September (1971), DOI 10.1109/TCT.1971.1083337.
[8] Strukov, D. B., G. S. Snider, D. R. Stewart, R. S. Williams. The missing memristor found. *Nature*, 06932, Vol. 453, pp. 80 – 83, (2008), DOI:10.1038/nature06932.
[9] Corinto, F., A. Ascoli. A boundary condition-based approach to the modeling of memristor nanostructures. *IEEE Transactions on circuits and systems – I, Regular papers*, Vol. 59, Issue 11, ISSN 1549 – 8328, pp. 2713 – 2726, November (2012).
[10] Walsh, A., Carley, R., Feely, O., Ascoli, A. Memristor circuit investigation through a new tutorial toolbox. *2013 European Conference on Circuit Theory and Design (ECCTD)*, *IEEE*, DOI: 10.1109/ECCTD.2013.6662261, pp. 1 – 4.
[11] Wen, S., Z. Zeng, T. Huang, X. Yu. Noise cancellation of memristive neural networks. *Neural Networks 60, Elsevier*, pp. 74 – 83, 2014, DOI: 10.1016/j.neunet.2014.07.014.
[12] Thomas, A. Memristor-Based Neural Networks. *Journal of Physics D: Applied Physics 46*, pp. 1 – 12, 2013, DOI: 10.1088/0022-3727/46/9/093001.
[13] Adhikari, S., C. Yang, H. Kim, L. Chua. Memristor Bridge Synapse-Based Neural Network and Its Learning. *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 23, 2012, pp. 1426 – 1435.
[14] The MathWorks, Inc. Neural Network Toolbox ™ User's Guide. MATLAB, Release 2009a.
[15] Chu, M., Kim, B., Park, S., Hwang, H., Jeon, M., Lee, B. H, Lee, B. G. Neuromorphic Hardware System for Visual Pattern Recognition With Memristor Array and CMOS Neuron. *IEEE Transactions on Industrial Electronics*, Vol. 62, No. 4, April 2015, DOI: 10.1109/TIE.2014.2356439, pp. 2410 – 2419.
[16] Hu, M., Li, H., Chen, Y., Wu, Q., Rose, G., Linderman, R. Memristor Crossbar-Based Neuromorphic Computing System: A Case Study. *IEEE Transactions on Neural Networks and Learning Systems,* Vol. 25, No. 10, October 2014, DOI: 10.1109/TNNLS.2013.2296777 pp. 1864 – 1878.
[17] Sheri, A., Hwang, H., Jeon, M., Lee, B. Neuromorphic Character Recognition System With Two PCMO Memristors as a Synapse. *IEEE Transactions on Industrial Electronics*, Vol. 61, No. 6, June 2014, pp. 2933 – 2941, DOI: 10.1109/TIE.2013.2275966.
[18] Cai, W., Ellinger, F., R. Tetzlaff. Neuronal Synapse as a Memristor: Modeling Pair- and Triplet-Based STDP Rule. *IEEE Trans. on Biomedical Circuits and Systems*, Vol. 9, No. 1, 2015, pp. 87 – 95, DOI: 10.1109/TBCAS.2014.2318012.