Development and Implementation of the LIRA Neural Classifier

Alejandro Vega, José Luis Pérez, Tetyana Baydyk, and Ernst Kussul

***Abstract*—Neural network is a tool in the solution of control problems. Thanks to their capacity for learning it is possible to train neural networks to recognize different patterns. The patterns involve diverse conditions of operation under which the system must be trained to be able to make the decisions to control the system or process. One of the methods of neural network simulation is digital design. We begin from a neuron design, and then we simulate the neural network that can be applied to the control process. The schematic design of logic circuits allows investigation of the circuit behavior and verification that the circuit fulfills the desired goals.**

***Index Terms*—LIRA neural classifier, logic circuits, neural networks, neuron.**

I. Introduction

This article is devoted to development and digital implementation of the LIRA neural classifier. The digital design is based on the boolean algebra [1]. It might be defined as a set of elements, operators and a number of axioms [2]. There are simplification methods of boolean expressions that allow us to reduce these expressions in a fast and easy way [3] – [5].

The design of an electronic model of a single neuron and the investigation of some of its dynamic behavior is a very interesting and important element in neural network development. It allows us to verify if the model complies with the objective set in neural network research. In particular, how the neuron model presents the dynamical behavior. The electronic neuron can commute between, at least, two different kinds of oscillatory behavior [6]. For example, the model is integrative as a leaky integrator below the threshold level and shoots when it reaches that level; then the neuron's potential performs in a similar way to an integrate and fire model. As a consequence of this, the neural response can be formed with different combinations of spikes and pulses [7].

The neuron model is a basic processing element of computational neural networks [8], [9]. Computational ability and biophysical reliability are two focuses of interest for researchers in these two different areas, and it's clear that new insights into either of these aspects would inevitably benefit both. The current generation of artificial neural networks found in both areas makes use of highly simplified models of neurons, such as summation–and–firing models [10]. Many researchers believe that the complicated dynamic behavior and advanced functions of a neural network system are primarily the result of collective behaviors of all involved neurons, and are less relevant for the operational detail of each processing element [11] – [13].

Several neural networks paradigms are being developed in CCADET, UNAM. They can be used to solve diverse problems. Some of the most interesting paradigms are RSC (*Random Subspace Classifier*), LIRA (*LImited Receptive Area*) and PCNC (*Permutative Coding Neural Classifier*) neural classifiers [14]. Image recognition tasks, for example, handwritten digits recognition, human face recognition, form and texture recognition in micromechanics are some problems which can be solved with neural classifiers [15] – [17].

II. LIRA Neural Classifier

In this article we will describe the LIRA neural classifier. The LIRA classifier is based on Rosenblatt perceptron principles [18]. Two variants of the LIRA classifier were proposed: LIRA_binary and LIRA_grayscale. The first one is used for recognition of binary (black-and-white) images and the second one is used for recognition of grayscale images [15]. Some changes in perceptron structure, training and recognition algorithms were made. The algorithm of LIRA classifier consists of the stages shown in Fig. 1. The LIRA classifier contains three layers of neurons. The first *S*-layer corresponds to the retina which, in technical terms, corresponds to the input image (Fig. 2). The second *A*-layer (associative layer) corresponds to the feature extraction subsystem. And the third *R*-layer corresponds to the system output; each neuron of this layer corresponds to one of the output classes. For example, in the handwritten digit recognition task this layer contains 10 neurons corresponding to digits 0,…, 9.

Manuscript received July 2, 2010. This work was supported by projects CONACYT 50231, PAPIIT IN110510-3, PAPIIT IN119610 and project of ICyTDF 332/2009.

Alejandro Vega is with the Center of Applied Science and Technological Development, National Autonomous University of Mexico (UNAM), México. D. F. 04510 México (Corresponding author Tel: (52) 55 56228602 ext. 1206, Fax: (52) 55 55500654, E-mail: alexandro_veg@yahoo.com).

José Luis Perez is with the Center of Applied Science and Technological Development, National Autonomous University of Mexico (UNAM), México. D. F. 04510 México (E-mail: pepito@aleph.cinstrum.unam.mx).

Tetyana Baydyk is with the Center of Applied Science and Technological Development, National Autonomous University of Mexico (UNAM), México. D. F. 04510 México (E-mail: tetyana.baydyk@ccadet.unam.mx).

Ernst Kussul is with the Center of Applied Science and Technological Development, National Autonomous University of Mexico (UNAM), México. D. F. 04510 México (E-mail: ernst.kussul@ccadet.unam.mx).

ISBN: 978-988-18210-0-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

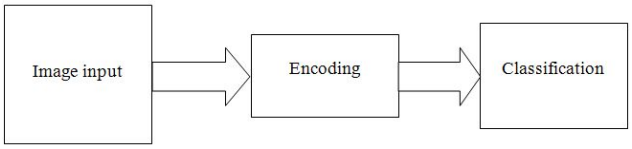

Fig. 1. Stages of the LIRA classifier algorithm

The connections between *S* and *A* layers are established using a random procedure and cannot be changed during the experiments with the neural classifier. They have the weights 0 or 1. Each neuron of the *A*-layer has connections with all neurons of the *R*-layer. Initially, the connection weights are set to 0. The weights are modified during the LIRA training. The rule of weights modification slightly different from Rosenblatt's one was used. The latest modifications are related to the rule of the winner selection in the output *R*-layer.

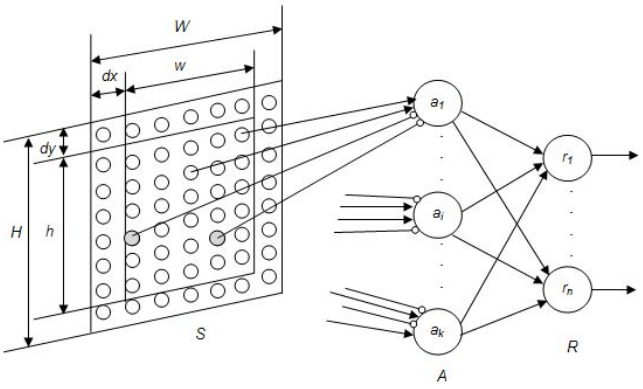

Fig. 2. LIRA classifier structure

The first stage is an image input. The image has a rectangular form with dimensions $H \times W$ (Fig. 2). The *S* layer corresponds to the image and it is the system input layer. This layer, also called retina, has a set of neurons connected to the associative *A*-layer. Each pixel of the input image corresponds to a neuron in the *S*-layer. It is necessary to enumerate all these neurons in *S* layer. The total number of them is $N = H \times W$. The $h \times w$ window is generated, which has random connections with the *A*-layer. A random selection is made from [1, *M*], where $M = h \times w$.

In Fig. 2 four random connections for one window of $h \times w$ to one *A*-layer neuron are shown. Among these four connections there are two connections of excitation (marked by an arrow) and two connections of inhibition (presented by a circle). The excitation connections increase the activity of the respective neuron, and the inhibition ones decrease this activity.

In this way we will generate all the connections between the *S* and *A* neurons. All windows may have different size ($h \neq w$), or a particular case can be applied, they may have the same size ($h = w$). They can be superimposed. A different criterion can be applied in the case of size and form of the windows generated in the retina.

In Fig. 2, the *dx* and *dy* distances are random numbers selected from [0, $W - w$] and [0, $H - h$], respectively.

To increase the speed of the neural network training we decided to simulate them with electronic circuits. Simulation of a digital neuron is the first step in the schematic design of neural networks. A description of a neuron model and its schematic design is presented below.

III. Implementation of the Neuron

Fig. 3 shows the neuron model. Two inputs groups are included: two *ON* neurons, and two *OFF* neurons; and a threshold *Th*, which controls the *S* output activity.

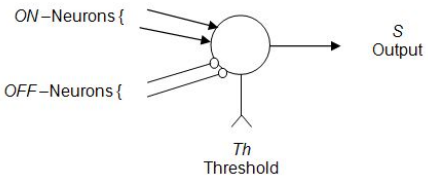

Fig. 3. Neuron model

Neuron digital implementation with Altera Max plus II software includes an adder designing to operate with excitation inputs, *ON* neurons, and another one to operate with inhibition inputs, *OFF* neurons. The adders use one bit, because the inputs to be added have 1 bit, (in accordance with the model shown in Fig. 3). Fig. 4 shows the design of the neuron with basic gates. For the *ON* neuron adder, an EX-OR gate is used (S_0 addition), and an AND gate for C_0 overflow, that is, $S_0 = O_0$ XOR O_1, and $C_0 = O_0$ AND O_1, where O_0, and O_1 are the two *ON* neuron inputs.

For *OFF* neurons the process is similar, so, summing up, $S_1 = O_2$ XOR O_3, and $C_1 = O_2$ AND O_3 are obtained, where O_2 and O_3 are *OFF* neuron inputs, and the gates array is also shown in Fig. 4. The design is the same, only the input and output subscripts of the circuit have changed.

To realize the inhibition process a subtracter is designed. The subtracter has two two-bit inputs. In Fig. 4 the first part of the subtracter is shown, S_0 minus S_1. $R_2 = S_0$ XOR S_1, and $C_2 = \hat{S_0}$ AND S_1, the "^" is a negation symbol. In the second part of the subtracter, C_2 is overflow. Fig. 4 also shows this part of the subtracter with carry in the subtraction process: $R_3 = C_0$ XOR (C_1 XOR C_2) and $C_3 = [\hat{C_0}$ AND (C_1 XOR C_2)] OR (C_1 AND C_2).

During the inhibition process a result is compared with threshold, *Th*. A comparator circuit is designed. R_3 and R_2 are subtracter outputs, which will be compared with the threshold value, denoted *Th*, compounded with Th_1 and Th_0, because *R* has a two-bit structure. This comparator has the *S* output, which will be high, $S = 1$, if the subtraction result is bigger than or equal to the threshold value, this means, $R \geq Th$. Otherwise, it will be zero. The result is: $S = [R_3$ AND (R_2 OR $\hat{Th_0}$)] OR [$\hat{Th_1}$ AND (R_2 OR R_3)] OR ($\hat{Th_1}$ AND $\hat{Th_0}$), the "^" symbol implies variable negation. In Fig. 4 the circuit that defines the comparison process between the resulting subtraction value and the threshold is shown.

The experiments with the neuron model demonstrated the possibility to implement the LIRA neural classifier with the Altera University Program Design Laboratory Package.

ISBN: 978-988-18210-0-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

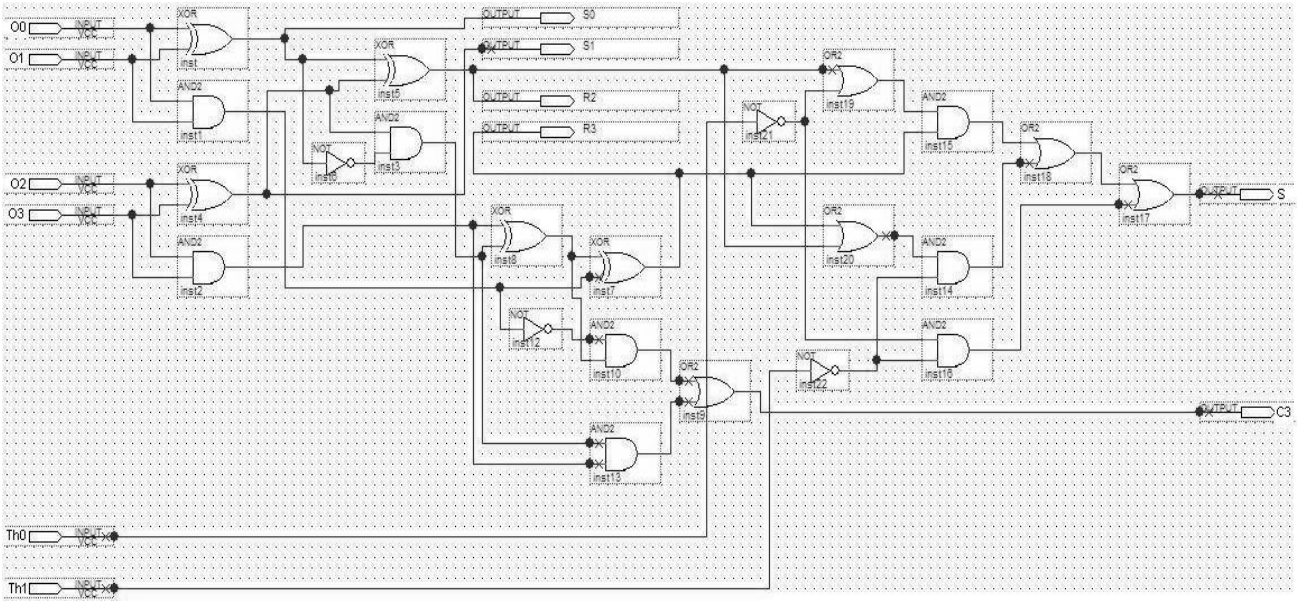

Fig. 4. Model design of the neuron

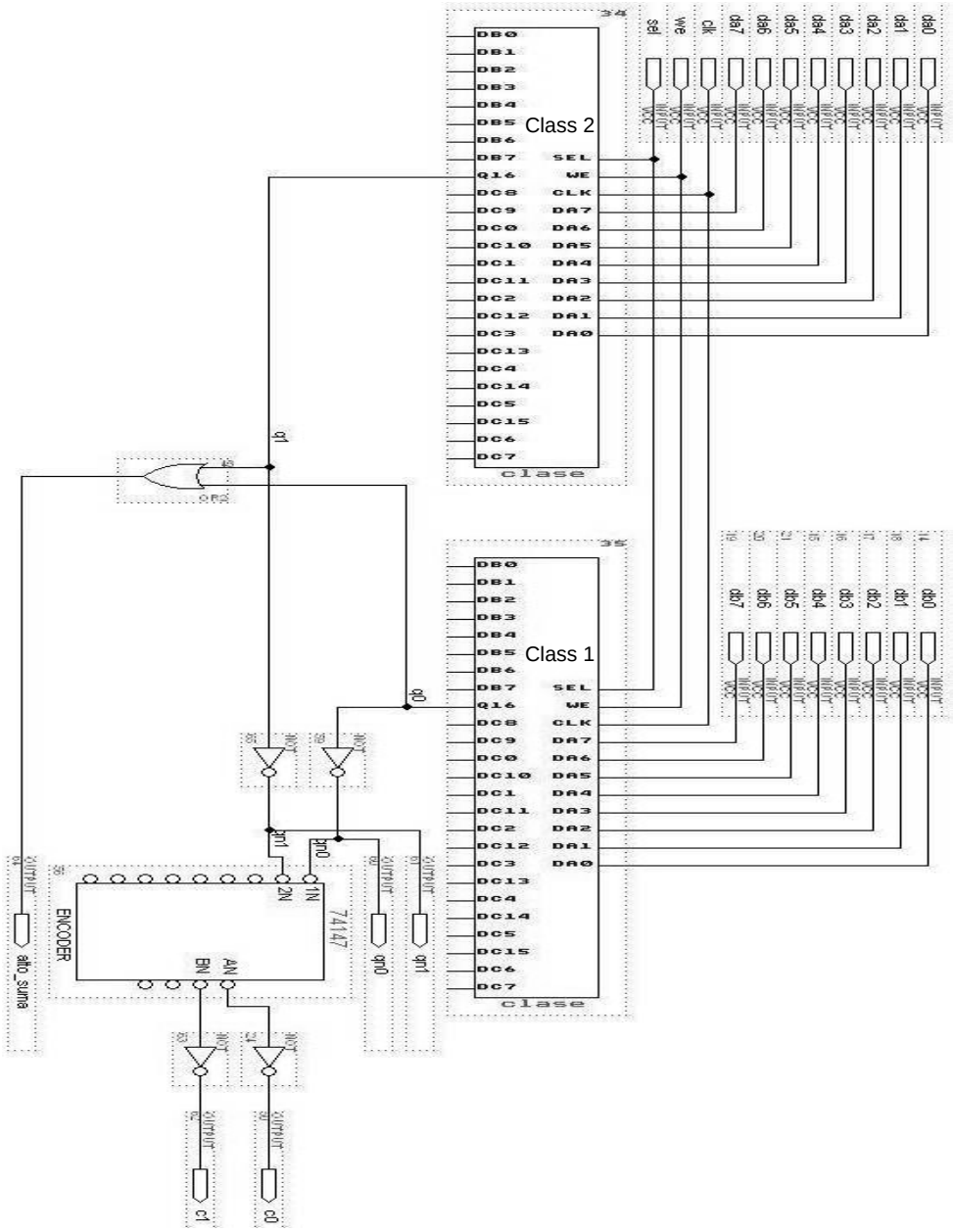

Fig. 5. LIRA for two classes

ISBN: 978-988-18210-0-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

IV. LIRA Neural Classifier Implementation for Two Classes

A simple version of LIRA structure for two classes with Altera was implemented. Fig. 5 shows the electronic circuit for the implementation of the two-class LIRA neural classifier. The circuit contains two structures. Every structure has 8-bit inputs, da[7..0] and db[7..0]. The circuit is designed for excitation calculation E_i, where i = 0, 1, and for the maximum excitation calculation E_{max}. Each structure simulates an output neuron, and maximum excitation shows the class under recognition.

The clock frequency is equaled to 25.125 MHz. The calculation process is parallel, that is why *clk* is applied to the both structures. When *we* = 1, data writing is in process, and blocks any operation, so zero data is shown in output. If *we* = 0, structures calculate the excitations, and determine the maximum excitation E_{max}. There are two data channels, one for data input and other for a "1". When *sel* = 0, excitation calculation is performed with input data, when *sel* = 1, calculation is performed with fixed data "1". The last case is for situation when there is no E_{max}, so it is necessary to force both classes to present an active E_i value. The calculation stops when any class is found.

To find the maximum value in the excitations E_i (Fig. 5) it is important to find significant bit (q16). Both classes have the same internal structure, so they have the same q16 output, they are named q0 and q1, and they will stop the addition process, when the maximum excitation value is obtained. It means, they show what neuron has the maximum excitation, pointing to the winner class; q0 represents class 1, and q1 represents class 2.

The ENCODER (Fig. 5) determines the structure with maximum excitation, showing the class number.

V. Results

The circuit of the LIRA neural classifier for two classes was simulated with the Altera University Program Design Laboratory Package. In Fig. 6 the Altera board during the first class recognition is presented.

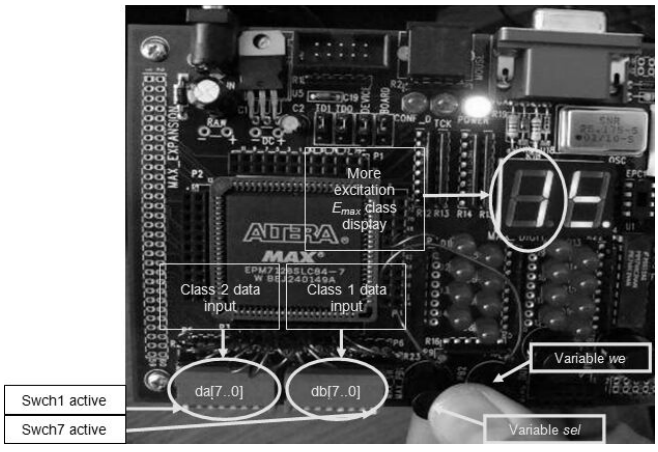

Fig. 6. The first class recognition

In Fig. 7 the second class recognition is presented.

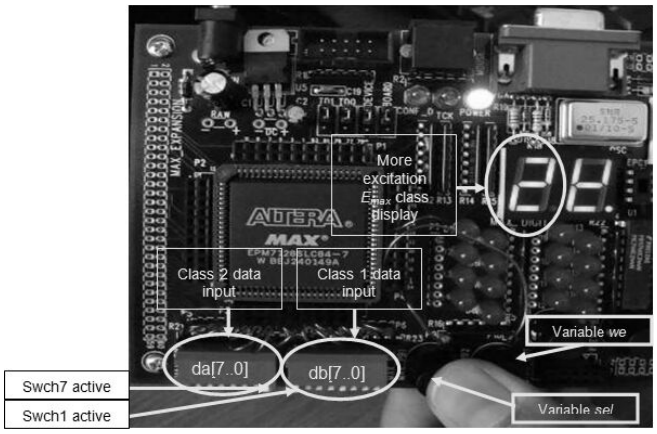

Fig. 7. The second class recognition

The results obtained during these experiments are explained. The simulations are presented in Fig. 8 and Fig. 9.

Fig. 8 shows db[7..0] data input with a constant value FF. The signal *sel* = 0. Data from data input da[7..0] is with constant value 00. The continuous addition of FF gives result bigger than the 16-bit adder size. This indicates a maximum value, and enables q0 pin, related to class 1. The ENCODER presents the class 1 as the class with maximum excitation.

Fig. 9 shows db[7..0] data input with a value 00 and da[7..0] has constant value FF. The signal *sel* = 0. The continuous addition of FF forces to an excessive result, bigger than the 16-bit adder size. This indicates a maximum value, and enables q1 pin, related to class 2. The ENCODER presents the class 2 as the class with maximum excitation.

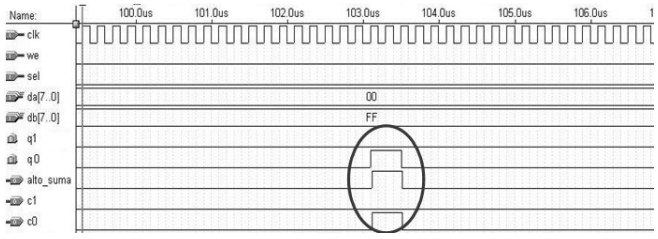

Fig. 8. Recognition, class 1

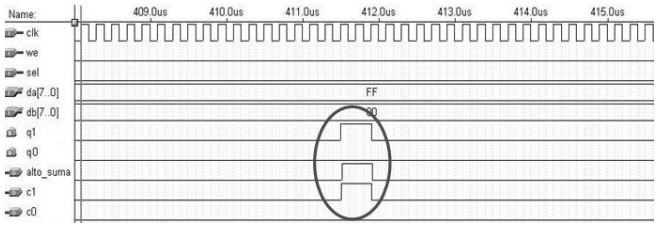

Fig. 9. Recognition, class 2

VI. Conclusion

The digital circuit of the neuron model was schematically designed and implemented. When the input signals are applied, this logic array adds them, and then inhibits them with the subtracter, applying comparison with the threshold. This threshold defines if the neuron is in inactive or active state. The software has allowed us to structure and to implement the logic circuit that simulates the neuron behavior. This step helps us to realize the implementation of the LIRA neural classifier.

ISBN: 978-988-18210-0-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

The digital electronic circuit of the two-class LIRA neural classifier was also schematically implemented. The simulations demonstrate the correct output for the both classes, when the input signals are applied. This digital electronic array calculates maximum excitation E_{max}, the winner neuron class. The same software has allowed us not only to implement the electronic schematic but also to simulate their behavior.

The contribution of this work is to demonstrate that the LIRA neural classifier can be implemented in a programmable logic device, and can be used in control systems based on image recognition.

REFERENCES

[1] G. Boole, *An Investigation of the Laws of Thought, on which are Founded the Mathematical Theories of Logic and Probabilities*. MacMillan and Co. London, 1854, 424 pp.

[2] C. E. Shannon, "A symbolic analysis of relay and switching circuits," *Transactions of the AIEE*, Vol. 57, No. 6, 1938, pp. 713-723.

[3] M. Karnaugh, "A map method for synthesis of combinational logic circuits," *Transactions of the AIEE*, *Communications and Electronics 72*, part I, 1953, pp. 593-599.

[4] W. V. Quine, "The problem of simplifying truth functions," *In American Mathematical Monthly*, vol. 59, No. 8, 1952, pp. 521-531.

[5] E. J. McCluskey Jr., "Minimization of boolean functions," *The Bell System Technical Journal*. Vol. 35, No. 6, 1956, pp. 1417-1444.

[6] Y. J. Lee, J. Lee, Y. B. Kim, J. Ayers, A. Volkovskii, A. Selverston, H. Abarbanel, and M. Rabinovich, "Low power real time electronic neuron VLSI design using subthreshold techniques," *IEEE Circuits and Systems*, vol. 4, 2004, pp. 744-747.

[7] A. Padrón, J. L. Pérez, A. Herrera, and R. Prieto, "Dynamical behavior of an electronic neuron of commutation," *In Proceedings of the Mexican international Conference on Artificial intelligence: Advances in Artificial intelligence*, April 11 – 14, 2000, *Lecture Notes In Computer Science*, vol. 1793, Springer-Verlag, London, pp. 338-349.

[8] W. S. McCulloch, and W. A. Pitts, "A logical calculus of the ideas imminent in nervous activity," *Bulletin of Mathematical Biophysics 5*, 1943, pp. 115-133.

[9] K. Shin'Ichiro, I. Makoto, and H. Nozomu, "Analog LSI neuron model inspired by biological excitable membrane," *Systems and Computers in Japan*, Volume 36, Issue 6, 2005, pp. 84 – 91.

[10] R. B. Szlavik, A. K. Bhuiyan, A. Carver, and F. Jenkins, "Neural-electronic inhibition simulated with a neuron model implemented in SPICE," *IEEE Engineering in Medicine and Biology*, vol. 14, no. 1, 2006, pp. 109-115.

[11] I. Requena, and M. Delgado, "A model of fuzzy neuron," *In Proceedings of the 2nd. International Conference on Fuzzy Logic Neural Networks*, (IIZUKA´90), Iizuka, Japan, July 20-24, 1990, pp. 13-26.

[12] C. Rasche, and R. J. Douglas, "An improved silicon neuron," *Analog Integrated Circuits and Signal Processing*, Volume 23, Number 3, 2000, pp. 227–236.

[13] J. L. Pérez, A. I. Miranda, and A. M. Garcés, "A neuron model with parabolic burst response," *In Advances in Artificial Intelligence and Engineering Cybernetics*, Vol X, 2003, pp. 6-10.

[14] T. Baidyk, E. Kussul, O. Makeyev, A. Caballero, L. Ruiz, G. Carrera, and G. Velasco, "Flat image recognition in the process of microdevice assembly," *Pattern Recognition Letters*, Vol.25, Issue 1, 2004, pp. 107-118.

[15] E. Kussul, and T. Baidyk, "Improved method of handwritten digit recognition tested on MNIST database," *Image and Vision Computing*, Vol.22, Issue 12, 2004, pp.971-981.

[16] E. Kussul, and T. Baidyk, "LIRA neural classifier for handwritten digit recognition and visual controlled microassembly," *Neurocomputing*, Vol. 69, Issue 16-18, 2006, pp.2227-2235.

[17] E. Kussul, T. Baidyk, D. Wunsch, O. Makeyev, and A. Martín, "Permutation coding technique for image recognition systems," *IEEE Transactions on Neural Networks*, Vol. 17/6, 2006, pp. 1566-1579.

[18] F. Rosenblatt, *Principles of Neurodynamics*. Spartan Books, New York, 1962, 215 pp.

ISBN: 978-988-18210-0-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)