Reconfigurable Self-Organizing NN Design Using Virtex FPGA

Janusz Starzyk and Yongtao Guo

School of Electrical Engineering and Computer Science
Ohio University, Athens, OH 45701, U.S.A.
Tel: 740-593-1580 Fax: 740-593-0007
Email: {starzyk, gyt} @bobcat.ent.ohiou.edu

Abstract—**In this paper, a self-organizing neural network model with entropy-based evaluator called EBE is proposed. An FPGA based design that implements the EBE model is presented. The PCI bus interface including DMA transfer is embedded into the design. 8-bit test data is fed into the design to verify the correctness of the algorithm and its FPGA implementation.**

Topic Area — No.5 : Algorithms and Optimization
and/or No.4: Reconfigurable Hardware Architectures

I. INTRODUCTION

Artificial neural networks (ANNs) are systems based on mathematical algorithms, which are derived from the field of neuroscience and are characterized by intensive arithmetic operations [1]. These networks display interesting features such as parallelism, classification, optimization, adaptation, generalization and associative memories [2,3]. To implement NN algorithm, custom hardware, although fast and compact, require significant non-recurring engineering (NRE) cost which makes NN difficult to implement. Fortunately, technological developments have substantially increased gate densities, making reconfigurable computers, based on field programmable gate arrays (FPGAs), an attractive alternative. FPGAs provide effective programmable resources for implementing self-organizing digital ANNs. They are readily available, reconfigurable and have smaller NRE costs– all important advantages for ANNs applications. ANNs have two phases of operations, the learning phase and the retrieve phase. During the learning phase, a flexible and space-efficient hardware of self-organizing neural classifiers is constructed using FPGA from the elementary modules and the processing functions evolve from connecting a number of modules in every layer. The self-organization architecture can structure itself to execute the algorithm and restructure itself during execution depending on results. So it can make the NN structure solve the different application tasks such as pattern recognition, image processing and so on. Using the training data, the structure defining decisions is obtained by the evaluator circuit using entropy-based algorithm [4].

Relative entropy represents the information we can extract from the training data. The algorithm within the evaluator circuit evaluates the training data and defines a self-adapting learning architecture mapping it to specific functional units. A hardware unit which computes the entropy based information is called Entropy-based Evaluator (EBE) which searches for the maximum mutual information using one-dimensional searching space. The learning algorithm searches the sample space in parallel and finds the locally optimal arithmetic and logic operation threshold for the input signal values. This learning process results in fewer connections and can provide a very high-speed classifier for many real-time image recognition and other machine learning-based applications.

The algorithm is first verified in Matlab at system level simulation. Then the Matlab bit level simulation results give us a realistic reference to hardware implementation at RTL (Register Transfer Level). First, a behavioral model of the EBE algorithm is implemented by using VHDL. VHDL is the name of the IEEE 1076 Hardware Description Language standard for very high-speed integrated circuits design [5]. Then the neural network organization and the learning algorithm are modified for easy implementation in Field Programmable Gate Arrays (FPGA).

The 8-bit length stochastic input signals are serially read into the input buffer via PCI bus by control module implemented with finite state machine. Counter and comparator are implemented with simple accumulator and shift register. Complex Logarithm-based entropy computing is obtained by a fully exploited look-up-based architecture of many FPGAs. The Look-up-table (LUT) input pointer is combined with a simple shift-add-based structure to obtain the entropy information with probability scaling. Different modules are connected using 8-bit data bus and synchronously operated under 32MHz PCI clock extracted by the control unit.

The hardware organization using FPGA has been modeled and simulated in VHDL. Leanardo, an RTL/logic synthesis tool from Mentor Graphics Corp., has been used to generate the gate level of the proposed structure. The Xilinx family (XCV800 Virtex FPGAs) is chosen as target technology [6]. The FPGA implementation performance results have been verified by efficient and fast classification of two experimental classes. Each class has three dimensions with 1000 normally distributed samples in each dimension with different mean values and variances.

In section II, the self-organization structure is introduced. Section III deals with the design methodology followed by an FPGA-based architecture for the algorithm. Section IV talks about the synthesis and implementation results and finally, a discussion is given in Section V.

II. SELF-ORGANIZING NN STRUCTURE

Self-organization is a transit process. In our self-organization model, the NN is a feed forward structure that can decide about its own interconnections and neuron operations. It has a prewired organization that contains a

number of identical processing blocks (called neurons), which are pseudorandomly connected to the input nodes and other neurons. In addition, the Entropy-based Evaluator (EBE) is utilized to select a proper operation and input selection for each neuron.

Entropy is a nonlinear function to represent information we can learn from unknown data. In the learning process, we learn some constraints on the probability distribution of the training data from their entropy. So we can choose a probability model that is optimum in some sense given this prior knowledge about the training data. Here we choose the entropy based information index to built the neural network structure in the learning process.

A. Information Index and Information Deficiency

In the learning process, the training set of signals is searched sequentially class by class to establish the optimum point (threshold) which best separates signals of the various training classes. The quality of the partition is measured by the entropy based information index defined as follows:

$$I = 1 - \frac{\Delta E}{E_{\max}}$$

where

$$\Delta E = -\sum_s \sum_c P_{sc} \log(P_{sc}) + \sum_s P_s \log(P_s)$$

and

$$E_{\max} = -\sum_c P_c \log(P_c)$$

here P_c, P_s, P_{sc} represent the probabilities of each class, attribute probability and joint probability respectively. The summation is performed with respect to all classes and two subspaces (for training samples in a given subspace that either satisfy or do not satisfy threshold respectively). The information index should be maximized to provide an optimum separation of the input training data. When the limit value of the information index equals to 1, the problem at hand is solved completely, i.e. the training data is correctly classified (separated into various classes) by the NN. In order to accumulate learning results from different subspaces we consider what is the amount of added learning and weight it against increased system complexity and resulting error of statistical learning. This is the case when a set of training data is obtained from a small subspace of the original space and it is related to less reliable statistics about the training data. So we also define the subspaces information deficiency as following to indicate how much knowledge must be gained to resolve the classification problem in the given subspace.

$$d_s = \frac{\Delta E_s}{E_{\max}} = \frac{\sum_s \sum_c P_{sc} \log(P_{sc}) - \sum_s P_s \log(P_s)}{\sum_c P_c \log(P_c)}$$

Our training objective is to find the vector configuration and threshold value which maximizes normalized information index I. The learning process is used to maximize classifying information for each class. Logarithm function used in entropy evaluation can be implemented in analog circuits owing to the nonlinear characteristics inherent in CMOS devices. In digital implementation, the entropy function can be approximated either by a lookup table (LUT) or by a direct calculation. The latter asks for many logic resources and a long delay time. In the former, the lookup table approach, the *Plog(P)* function-value associated with each probability-value is stored in the memory and *P* is used as an address to the lookup table. Many EBE hardware unit will be used to support the organization and local optimization of evolved learning structure. Therefore, each of them should be simple and use small design area.

B. EBE Analysis and Simulation Results

In order to simplify hardware used for EBE, an effect of round off errors on the accuracy of the resulting information index is considered. The first test performed is to determine the dependence of the information index error on the number of units used to represent them. For simple verification, we use two classes training data with three dimensions per class. In every dimension, there are one-thousand normally distributed random values with different mean value and variance.

The EBE algorithm is first verified by Matlab simulation in both behavioral and structural level. The simulation results obtained in structural levels are shown in Fig. 1. In this example, two one thousand, one-dimensional and normally-distributed points with different mean values and variances are taken as training data. In the behavioral simulation, we use 8-bit widths to represent the input analog data and set threshold searching step to be maximum quantification noise.

Using the results obtained in behavioral simulation, approximation error effects were analyzed and generalized structural model was developed. In the structural level simulation, we utilize 5-bit width input pointer to address the *Plog(P)* lookup table (LUT) and to cope with the induced noise in the quantification process of the training data. As seen in Fig.1, these hardware simplification and approximation in the process of entropy calculation do not sacrifice the necessary classification information compared to the behavioral simulation results.

III. HARDWARE IMPLEMENTATION

A. Design Methodology

A top-down design methodology was adopted. A high-level VHDL model [7] for the circuits was generated. The logic was partitioned. Each part was re-described in a lower level description (RTL) required for the circuit synthesis, optimization and mapping to the specific technology by assigning current FPGA family and device. The resulting optimized circuit description was verified through extensive simulation after which the layout was created (Layout synthesis) and finally, on chip verification was executed by using C++ programming to connect PCI bus to the design ports and to test the design.

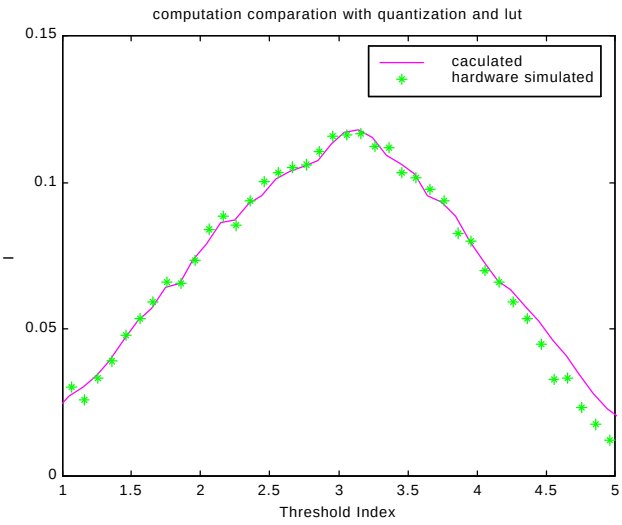

Fig.1 Structural Simulation

B. FPGA-based Architecture

The Xilinx Virtex XCV800 was adopted in our study as the Virtex series offers the improved architecture, high gate density and connecting line density. The density of XCV800 is 888,439 equivalent gates and it consists of a 56 x 84 grid of configurable logic blocks. Global Routing resources distribute clocks and other signals with very high fanout throughout the device. Some classes of signal require dedicated routing resources referred to as primary global and secondary local clock routing resources. In the speed grad-6 XCV800, Global Clock input to output maximum delay is 4.9 ns. Interconnection delay increases with increasing fanout and routing distance. The Virtex CLBs structure combines two LUTs and referring to LUT is useful for EBE with the LUT's flexible function implementation ability. It also has a fast propagating adder feature with dedicated circuitry for the computation and interconnection of carries. The fast propagate feature provides an implementation with the least delay and small design area. In our design, we embed PCI bus interface module including DMA for fast data IO and easy debugging with software. The system architecture is shown in Fig.2. In particular, Fig.2 illustrates the EBE hardware model which is mainly based on a:
Memory circuit unit (LUT) which implements the *Plog(P)* function.

- Comparator unit using a fast propagating carry feature to compare the current maximum entropy index with the calculated entropy from the entropy calculating unit (ECU).
- Two registers that are used to store the maximum entropy index and its corresponding threshold in the process.
- ECU which can produce the 5-bit access pointer for data acquirement from LUT, calculate the current information entropy and send the current entropy and threshold to the comparator unit.

In the four units which comprise the EBE module, ECU is the main block organized as shown in Fig.3. Other components are used for control, interface, monitor and so on.

- Control unit produces the control signals for the whole EBE including system clock, state transfer signals, handshake signals and so on.
- MUX and DMUX are used for parallel process of the multi-dimensional data in the input classes.
- Display unit implements the online monitor for the data transfer.
- EBE interface is used as interface between FIFO control unit, PCI bus and EBE for rapid data transfer and easy online system debugging.
- PCI interface core and FIFO unit satisfy PCI 2.0 specification.

C. VHDL Design and Simulation

We use VHDL to describe a digital system at the behavioral level so that we can simulate the system to check out the algorithm used and to make sure that the sequences of operations are correct. After verification the correctness of the algorithm and adaptability of the hardware implementation (Shown in Fig. 4), we add the PCI bus interface modules into the design and organize them as a hierarchical structure. The first level in hierarchy is the PCI interface includes DMA, FIFO etc.

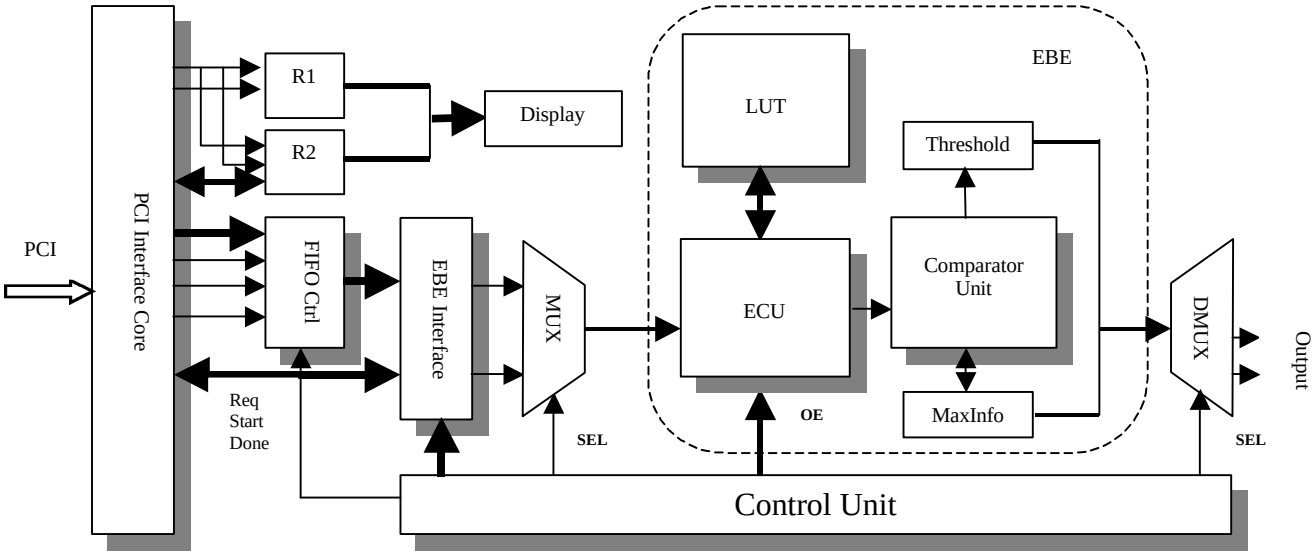

Fig.2 FPGA-based Architecture

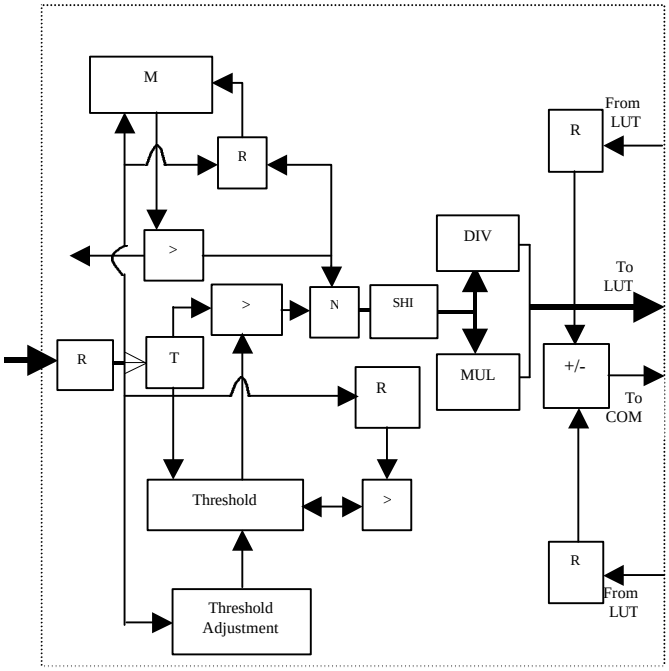

Fig.3 Entropy Calculating Unit

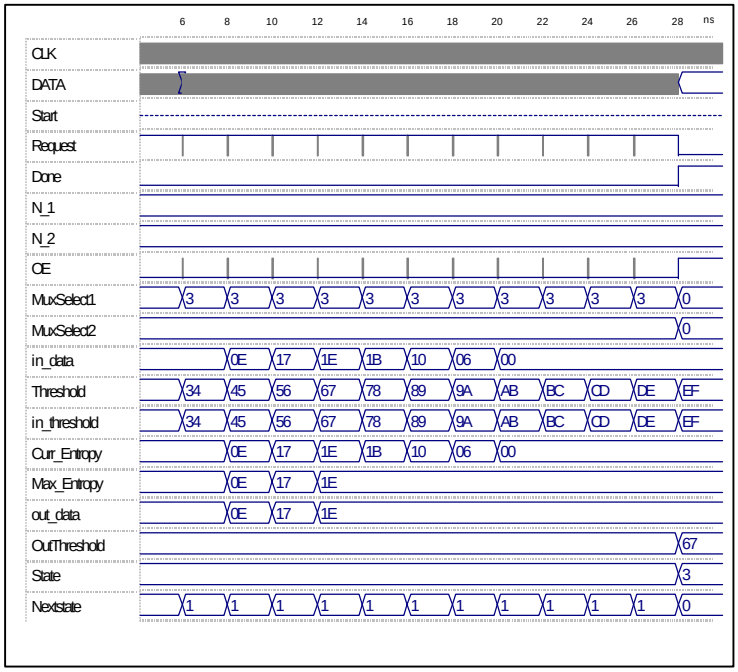

Fig. 4 VHDL Simulation at RTL

The second level is the EBE supporting module group and includes controller, MUX, DMUX and interface to PCI and calculating module. The last level is the core EBE module which is divided into calculating unit, LUT, comparator and registers. In the VHDL description in RTL level, we divide the module according to its hardware function division as seen in Fig.2. The searching threshold step, maximum threshold and data width are generic and can be configured easily.

The *Plog(P)* function is implemented by the ROM LUT with 5-bit width address. Other units adopt 8-bit data flow. The outer modules like PCI interface, FIFO and so on are linked to EBE module by 32-bit data bus. In the process of simulation, we use the lowest 10 bits as the data channel *I* from Class *I* and the upper 10 bits as the data channel *II* from Class *II*. From the simulation results, we can see that the data are transferred and controlled by the signals-Request, Start, Done, OE and current state.

As seen in Fig.2, these interface and control signals are transferred between EBE interface and PCI and between Control unit and other modules. After the current threshold reaches the maximum threshold set by the generic parameter, the Done signal will be set to High and the simulation process will be over. The last output threshold will be the classification threshold for the current dimension of the classes. In the EBE calculating process, the input data will be resended once the threshold is updated by the generic threshold step parameter while the signal Request to the interface model will be set to low. The Request, Done, Start, OE are also used as the handshake signal groups for the synchronized work of the whole hardware module. The simulation results are obtained by using Aldec-HDL simulator [8].

IV. SYNTHESIS AND PERFORMANCE

Most of units including PCI bus interface are synthesized using logic synthesis tool, Leanardo yielding the gate level structure of the full EBE model. The logic synthesis tool starts with two kinds of information: the RTL specification given in VHDL and a functional unit library, which can include complex functional units. The RTL description accesses these functional blocks through VHDL procedure calls. For each procedure or function used, the library must induce at least one functional unit able to execute the corresponding operation. The synthesized results are tested using commercially-available FPGA board (Nallatech Ballynuey board [9]) provided by Nallatech Inc. UK. The Ballynuey board is a PCI compatible expansion board that can be used via a PCI-compatible PC. The host PC stores all configuration information and collected data. And the configuration data can be downloaded to the Virtex FPGA board via PCI bus. Part utilizations for the PCI interface and EBE module were respectively 385 slices and 96 slices. The following table shows the synthesis report of the design and Fig. 5 shows the synthesized schematic and Virtex floorplan. We use VC++ as the software debugging tool to test the circuits and the design has been verified by functional level simulation and gate level simulation.

Vendor: Xilinx
Family: VIRTEX
Device: V800BG432
Speed: -4

Number of External GCLKIOBs 1 out of 4 25%

Number of External IOBs	47 out of 316	14%	Number of flip-flops:	336
Number of BLOCKRAMs	4 out of 28	14%	---	
Number of SLICEs	463 out of 9408	4 %	Minimum period: 24.838ns	
Number of DLLs	1 out of 4	25%	Maximum frequency: 40.261MHz	
Number of GCLKs	1 out of 4	25%	Total equivalent gate count for design: 88,186	
Number of TBUFs	256 out of 9632	2%	Additional JTAG gate count for IOBs: 2,304	

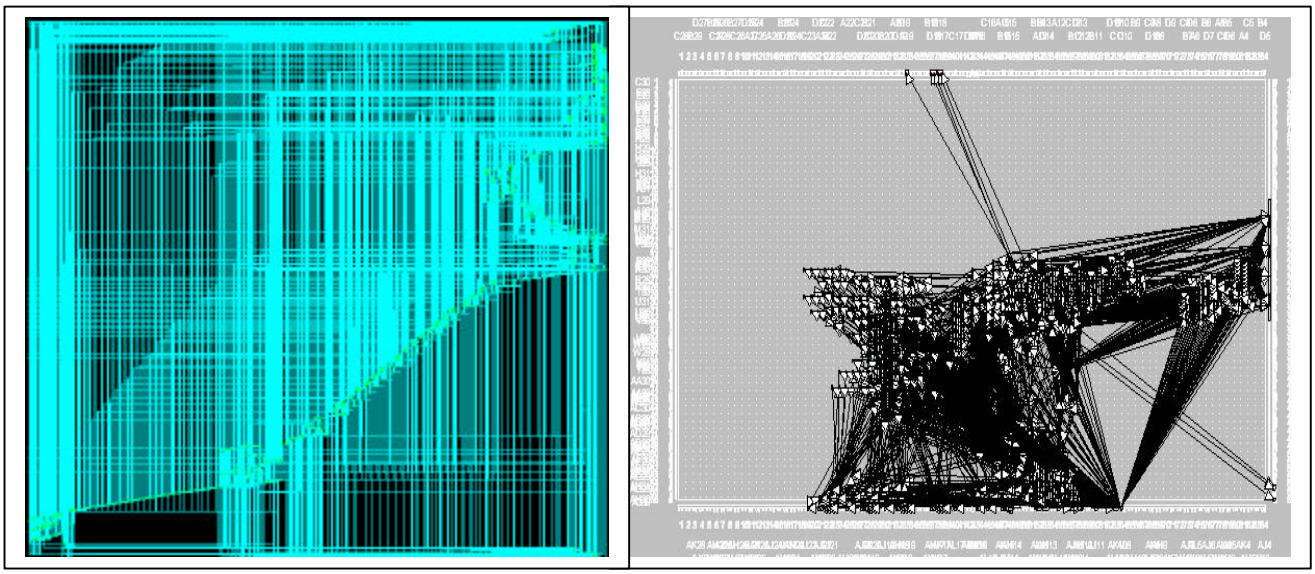

Fig.5 Synthesized Schematic & the Corresponding Floorplan in Virtex

As can be seen from the synthesized design, many EBE units can be implemented on a single Virtex chip together with the self-organizing neural network structure. Thus the entire classifier can be incorporated in this technology.

V. SUMMARY AND CONCLUSION

In the paper, we have presented an algorithm for digital implementation of ANNs based on system entropy. The developed models have been verified by VHDL simulation results. We use behavioral level to validate if the selected bit widths for internal and external signals are sufficient for achieving a required computation precision. In classification area, the necessary calculation accuracy varies by application. The low precision can simplify the hardware implementation complexity and speedup the performance. The EBE module, consisting of calculating unit, a memory unit and a few digital components, has been modeled and simulated in VHDL. Experimental results show that the obtained classification of the training data obtained by behavioral VHDL model match closely with that anticipated from the analysis results. Our next objective is to construct a parallel self-organizing neural network, verify hardware training phase and test the approach in the real world applications.

Another important development we are searching for is to use analog circuits to implement the algorithm. The Logarithm-based non-linear function can be easily implemented by the non-linear characteristics of analog circuit in a small design area [10]. Higher speed, smaller area and power dissipation of analog circuits constitute a potentially powerful improvement over digital circuits. This will be stated in the next phase of our research.

References

[1] S.Titri, H.Boumeridja, D.Lazib, N.Izeboudjen. “A Reuse Oriented Design Methodology for Artificial Neural Networks Implementation”. IEEE,1999.

[2] Martin T. Hagan, Howard B. Demuth, Mark Beale. “Neural Network Design”. PWS Publishing Company. 1995.

[3] S.Y.Kung.“Digital Neural Networks”. PTR Prentice Hall, 1993.

[4] J. A. Starzyk and J. Pang , “Evolvable binary artificial neural network for data classification.”, The 2000 Int. Conf. on Parallel and Distributed Processing Techniques and Applications, (Las Vegas, NV, June 2000).

[5] K.C.Chang, “Digital Design and Modeling with VHDL and Synthesis”, IEEE Computer Society Press,1997.

[6] Xilinx, “The Programmable Logic Data Book”, San Jose. 1993.

[7] S.S.Erdongan, Abdul Wahab, T. H. Hong. “VHDL Modeling and Simulation of the Back-Propagation Algorithm and its Mapping to the PM”. IEEE 1993 Custom Integrated Circuits conference.

[8] Aldec, “Aldec-HDL™ Series User Guide Version 4.1”, August 2000.

[9]“BALLYNUEY 2 VIRTEX PCI CARD USERS GUIDE”. Nallatech Ltd.1993-1999.

[10] Carver Mead “Analog VLSI AND NEURAL SYSTEMS”. ADDISON-WESLEY Publishing Company, 1989.