A SCALABLE DIGITAL ARCHITECTURE OF A KOHONEN NEURAL NETWORK

Andrés E. Valencia, Jorge A. Peña and Mauricio Vanegas.

Universidad Pontificia Bolivariana,
Medellín, Colombia
andrez_valencia@yahoo.com, jorge.pena@alari.ch, mvanegas@upb.edu.co

ABSTRACT

Kohonen self-organizing feature maps are unsupervised learning neural networks that categorize or classify data. Efficient hardware implementation of such neural networks requires the definition of a certain number of simplifications to the original algorithm. In particular, multiplications should be avoided by means of simplifications in the distance metric, the neighborhood function and the learning parameter values.
In many applications, a scalable solution becomes necessary due to the limited memory resources available in many embedded platforms. In this paper, a scalable Kohonen map called Local Winner-Take- All (LWTA) design with Minkowski norm L_∞ and an exponential neighboring function is defined, and its hardware architecture is presented. The scalability of the net is achieved via a Local Winner –Take- All approach. Results of VHDL simulations as well as synthesis on an FPGA of the proposed architecture demonstrate satisfactory functionality of such architecture.

1. INTRODUCTION

Artificial neural networks are parallel computational models comprised of densely interconnected adaptive processing units (neurons), able to learn a static map (supervised learning) or to classify and categorize the input space (unsupervised learning) from an input data [1]. A Kohonen's self-organizing feature map (Kohonen's SOFM) is an artificial neural network of unsupervised learning that captures the topology and probability distribution of input data [2].

Most of the neural networks applications have used simulations on conventional single-processor machines. The possibility of parallel processing and short operation times has encouraged the implementation of hardware neural networks [3], [4], [5], [6]. Kohonen's SOFM has not been the exception, and there are several implementations in analog [7] as well as digital circuits [8] [9]. Generally, digital implementations have been more successful because of their lesser vulnerability to noise and their higher scale of integration when is compared with their analog counterparts, additionally several laboratories currently work strongly on reconfigurable circuits in which this kind of digital architectures will be suitable[10].

In many applications, the data load can increase at an incredible rate and overcome the resources of the system. Moreover due to the limited memory resources available in many embedded platforms, such as mobile terminals [11], a scalable solution becomes necessary.

In [12] we proposed a very simple but efficient digital architecture of a Kohonen's SOFM. In this work, we add a new feature to this design in order to make it more easily scalable. The idea is to be able to construct a larger map by connecting similar chips (e.g. FPGAs implementing a certain number of interconnected neurons) without the necessity to change the whole architecture.

2. KOHONEN'S SELF ORGANIZING FEATURE MAPS

A Kohonen's self-organizing feature map is an unsupervised learning neural net that captures the topology and probability distribution of input data.
Its architecture consists of an array of units or neurons with a fixed position Ri within the map and a variable n-dimensional weight Wi , where *n* is the dimensionality of input patterns.

The weights of the neurons are updated each time a new input pattern is presented. The magnitude of the change of the weight of the neuron depends on the topological proximity to the winner neuron, whose index i* is given by:

$$i^* = \arg_i \min d_i(w_i, x^k)$$

When the winner neuron is found, the weight of the i-th neuron is updated according to:

$$w_i = w_{io} + \Delta w_i = w_{io} + \rho \cdot \Phi(r_i, r_{i^*})(x^k - w_{io})$$

where ρ is the learning rate and $\Phi(r_i, r_{i^*})$ is a neighborhood function.

3. THE PROPOSED KOHONEN'S SOFM

As described in [12], the LWTA design is based on the idea to simplify the algorithm of self organizing feature maps due to hardware aspects in order to minimize the necessary chip area and thus to maximize the number of processing unit per FPGA.

First the Chessboard distance is used instead of the Euclidean distance.

$$L_\infty(x, y) = \max_i | x_i - y_i |$$

where $L_\infty(x,y)$ is the distance metric between vectors x and y and x_i (y_i) is the i-th component of the vector x (y).

Therefore no multipliers are necessary to calculate the distance. Secondly the values of the learning rate are restricted to the set {1, ½, ¼ ... $(½)^\alpha$ }. Finally a discrete exponential neighborhood function is defined. With the proposed simplifications we can reformulate the weight update formula:

$$\Delta w_i = \begin{cases} \left(\frac{1}{2}\right)^{\alpha} (x^k - w_{io}) & si\ i = i^* \\ \left(\frac{1}{2}\right)^{\alpha + d(r_i, r_{i^*}) + \beta} (x^k - w_{io}) & si\ i \neq i^* \end{cases}$$

Therefore, Δw_i is obtained simply by multiplying (x^k -w_{i0}) by a power of ½, which can be easily implemented with a shifter, avoiding the necessity of multipliers (expensive logic) in the circuit.

4. HARDWARE DESIGN

A one-dimensional Kohonen map of 10 units is designed; the system is capable of handling a two-dimensional input patter with 8-bit resolution.

4.1. The Network

The network is designed to work in parallel and to optimize execution speed. Each neuron is defined as a building block that processes data in an independently fashion. The proper behavior of the net as an aggregation of these building blocks is guaranteed by one "global" control block, the parameter scheduler. Consequently, when several FPGAs are connected in the LWTA design, the learning rate signal, α, the neighborhood width signal, β, and the reset signal must be connected as well as the signals downout and downin with the signals upin and upout of the new FPGA (Fig. 1).

4.2. The Neuron

A Kohonen's neuron is composed of a Chessboard distance calculation block and a weight update block (Fig. 2). Each neuron includes four 8-bit and three 4-bit adders, one comparator and two shifters.

In order to assure a proper synchronization of the digital neuron, the Chessboard distance calculation block operates on falling edges and the weight updating block on rising edges of a clock signal.

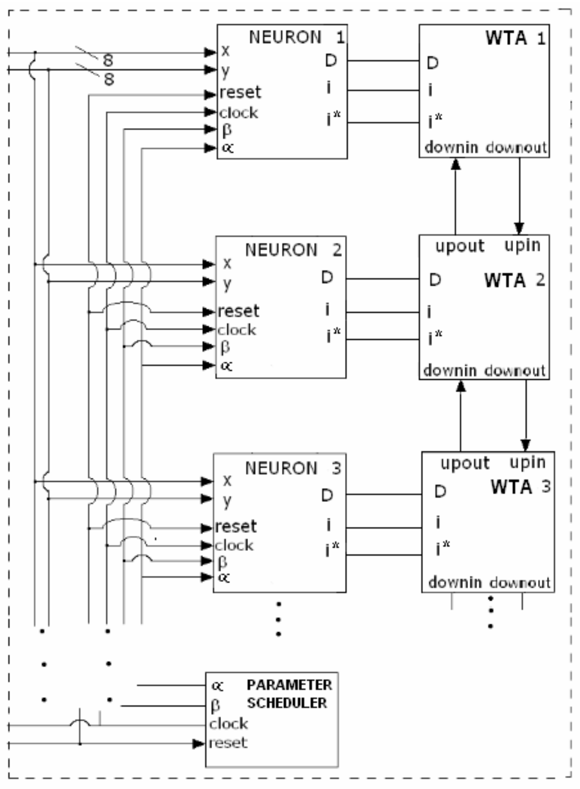

Fig. 1. Network architecture of the LWTA design.

The Parameter Scheduler

The parameter scheduler updates the learning rate parameter α and the neighborhood width parameter β incrementing them from an initial value of 0 to a final value determined beforehand. Basically, this block consists of two counters (one for each parameter) registers and a combinatorial logic to determine the set of time steps at which the parameters will be incremented.

4.4. The Winner -Take- All (WTA) Block

The WTA blocks in the LWTA design has two tasks. First, when reset is pressed, the neuron 1 sends a “1”, the neuron’s position in the map, to the neuron 2 throughout signal downout. Afterwards, the neuron 2 sends a “2” throughout signal downout to neuron 3, and so on, until all the neurons know their position in the map (Fig. 4a).

The second task of the WTA block is to find the minimum of the distances between all the neurons and the input pattern. Each neuron calculates the distance, D_j, on a falling edge, afterward the neuron’s j WTA block send the distance D_j and the position j to the adjacent neurons throughout signals upout and downout, then the WTA block of the neurons j+1 and j-1 receive the distance and the position throughout the signals upin and downin, and compares the distance with the proper value (Fig. 3).

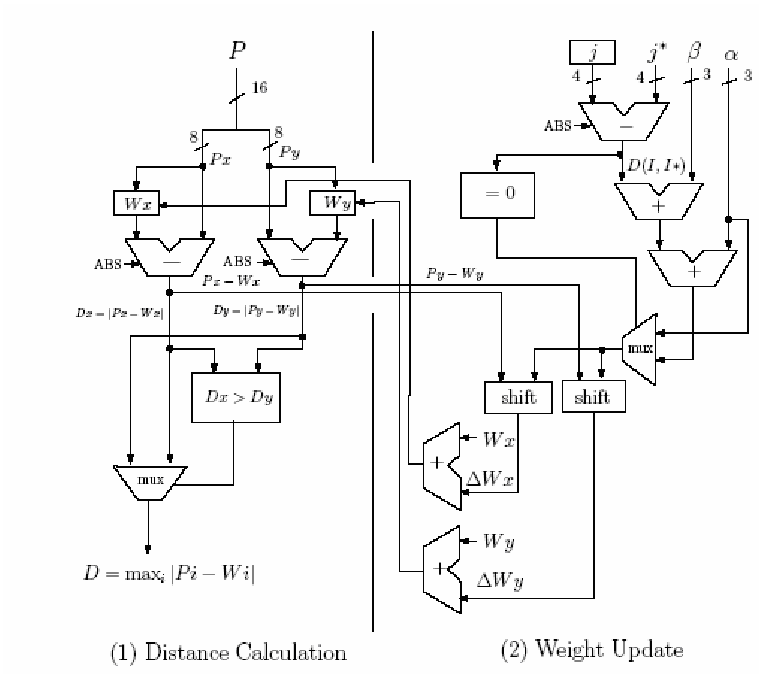

Fig. 2. Neuron’s architecture

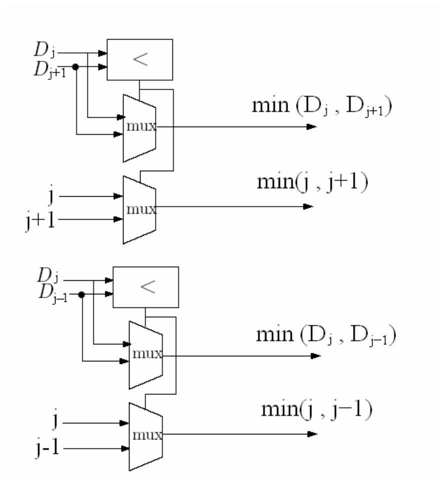

Fig. 3. Winner -take- all block of LWTA design.

For example, if neuron 4 has a distance of 100, neuron 5 has a distance of 176 and neuron 3 has a distance of 65 (Fig. 4b), then the output of the WTA blocks are 4 and 100 (100 < 176) and 3 and 65 (65 < 100) (Fig. 4c).

Note that the signal downin is the concatenation of j-1 and D_{j-1} and signal upin is the concatenation of j+1 and D_{j+1} (Fig. 3).

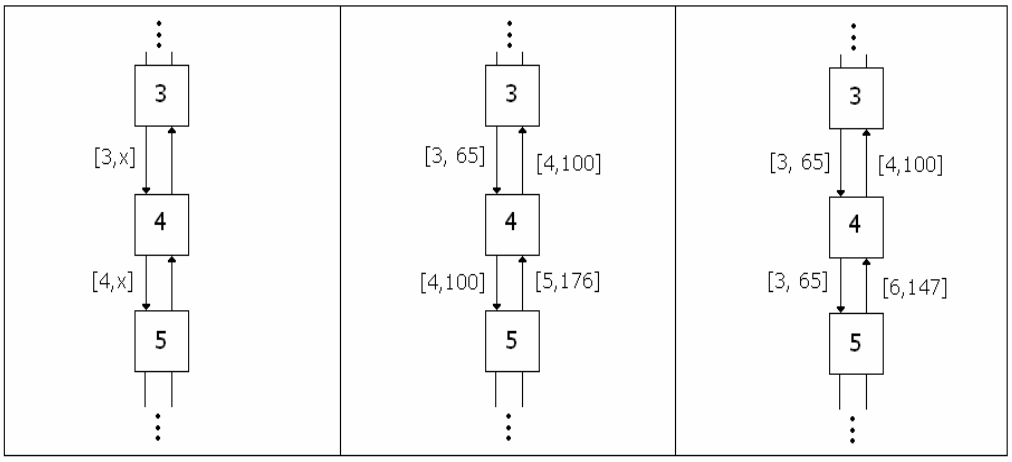

Fig. 4. Behavior of the LWTA design (a) in reset (b) at the beginning of the comparison (c) at the end of the comparison.

5. EXPERIMENTAL SETUP AND RESULTS

The experimental setup consists of two parts: the VHDL simulation of a Kohonen map for a simple clustering task and the synthesis of the net on an FPGA.

5.1. Behavioral Simulation

The proposed hardware was described in VHDL and then simulated using the Modelsim Simulator XE II 5.7g of Mentor Graphics [13] for two simple clustering tasks.

The first input data consisted of 1000 points randomly generated in the xy plane between lines with slopes 1 and intersections of +25 and -25.
The net was trained for 18000 iterations (time steps). The parameters α and β were both initialized at zero, while the neuron weights were initialized at (125, 125). The parameter α was incremented by one at time steps 4000, 8000 and 12000, and the parameter β at time steps 2000, 6000, 10000 and 14000.

Fig. 5 shows graphs of the input data and the weight vectors for different phases of the algorithm. It can be seen how the units arrange themselves so as to follow the probability distribution of input vectors. Furthermore, the weight vectors are finally ordered according to their mutual similarity, so that neurons close to each other in the linear array correspond to neurons with close weights in the input pattern space, as must be the case of a well trained Kohonen net.

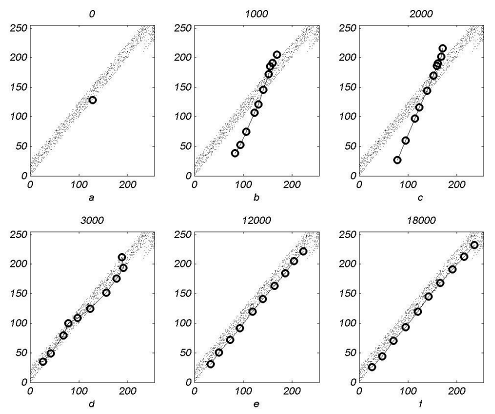

Fig. 5. Input data for a simple clustering task (small dots) and weight vectors (circles) in the input space: (a) Iteration 0, (b) Iteration 1000, (c) Iteration 2000, (d) Iteration 3000, (e) Iteration 12000, and (f) Iteration 18000

The histogram of Fig. 6 shows for each neuron the number of times it became the winner unit of the net. At the end of the run, each neuron has almost the same chance to win, which assures that the neurons have covered the input space following the probability distribution of the input data.

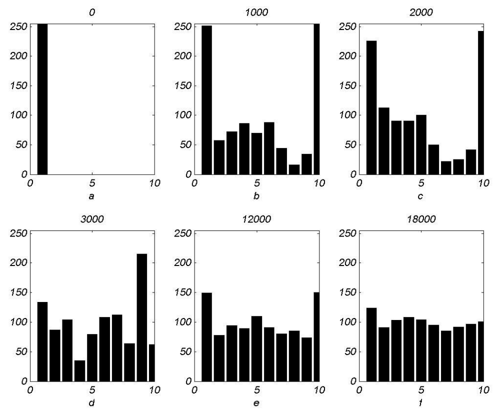

Fig. 6. Histogram of winning events: (a) Iteration 0, (b) Iteration 1000, (c) Iteration 2000, (d) Iteration 3000, (e) Iteration 12000, (f) Iteration 18000

The second input data is an Archimedean spiral with a polar equation given by:

$$r = \alpha\theta^{1/n}$$

Where r is the radial distance, θ is the polar angle and n is the constant which determines how tightly the spiral is "wrapped".

The input data consisted of 1000 points generated with a radial distance of 30 and a constant of 2. The net was trained for 18000 iterations. The parameters α and β were both initialized at zero, while the neuron weights were initialized at (125, 125). The parameter α was incremented by one at time steps 4000, 8000 and 12000, and the parameter β at time steps 2000, 6000, 10000 and 14000.

Fig. 7 shows graphs of the input data and the weight vectors for different phases of the algorithm.

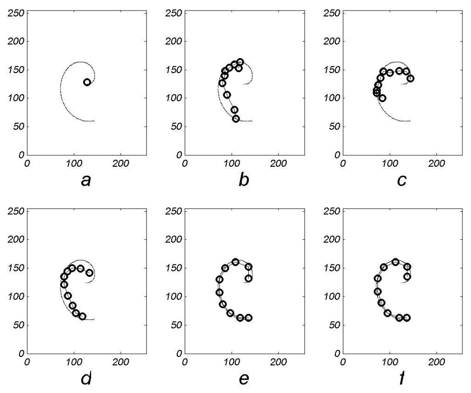

Fig. 7. Input data for a simple clustering task (small dots) and weight vectors (circles) in the input space: (a) Iteration 0, (b) Iteration 1000, (c) Iteration 2000, (d) Iteration 3000, (e) Iteration 12000, and (f) Iteration 18000

The histogram of Fig. 8 shows for each neuron the number of times it became the winner unit of the net.

Note that a common property in the SOFMs is a border aberration effect that causes a slight contraction of the map and a higher density of weights at the borders. This aberration is cause by the "pulling" by the units in the map thus the asymmetric behavior of the histogram.

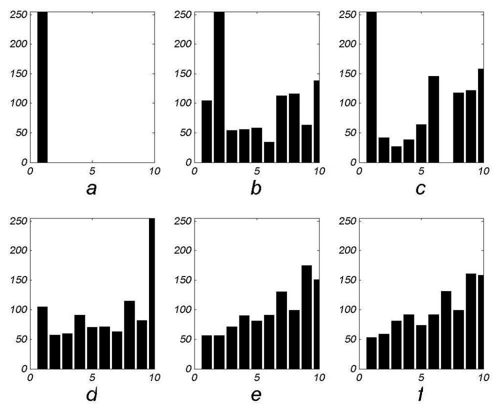

Fig. 8. Histogram of winning events: (a) Iteration 0, (b) Iteration 1000, (c) Iteration 2000, (d) Iteration 3000, (e) Iteration 12000, (f) Iteration 18000

5.2. Synthesis

The description of a one-dimensional LWTA design composed of 10 units with chessboard metric and exponential neighboring function was synthesized on an FPGA in order to determine the suitability of a real implementation of the proposed hardware. An FPGA is an array of logic cells whose functionality and interconnection can be programmed by a configuration bit stream [14]. We used a Spartan III xc3s200ft256 from Xilinx Corp. [15] which has a maximum capacity of implementing 200000 logic gates. This FPGA has 24 x 20 configurable logic blocks (CLBs). The Kohonen map used 1357 slices (70% of the whole FPGA). It must be said, however, that no attempt was made to optimize the synthesis.
The maximal xc3s200's pin-to-pin delay was 5.5 ns. Given that the processing time per input vector is one clock cycle and that both, the rising and the falling edge of the clock signal are being used, the system could process a new input vector every 11 ns.

6. A Comparison of resources of both designs

The LWTA design and the GWTA design proposed in [12] are compared by measuring the chip area (slices, flip flop slices and look up tables) and speed execution (maximum pin delay).

Designs	GWTA	LWTA
Slices	1126	1357
F.F. Slices	440	545
LUTs	1691	2141
Maximum pin delay (nseg)	5.454	5.5

Although both designs process input data in one clock cycle, the LWTA design is not as efficient as the GTWA design in term of silicon area and speed execution. This difference is due to the conception of the WTA block.

The GWTA design finds the minimum distance between the weights of the neurons and the input pattern by defining a global block, the WTA block (Fig, 9). However, when several FPGAs are connected, each FPGA has a part of the global map thus the existence of this block creates a problem.

Therefore, in a Kohonen's SOFM of *n* neurons, the WTA block of the GWTA design has *n-1* multiplexors and *n-1* comparators (Fig. 9), however in the LWTA design, the WTA block has 2 multiplexors and 2 comparators per neuron (Fig. 4) for a total of *2*n* multiplexors and *2*n* comparators.

In the GWTA design, the WTA global is composed of comparators and multiplexors arranged in a cascade configuration (Fig. 9).

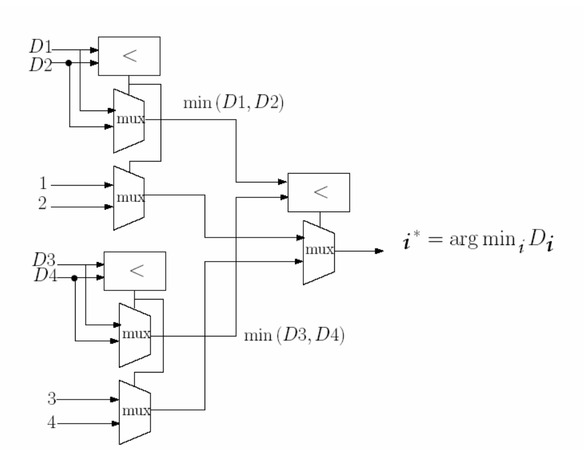

Fig. 9. Winner take all block of GWTA design. For the sake of simplicity, a 4-neuron map is assumed.

Nevertheless The Spartan x3c400 can have 30 neurons of the GWTA design but the LWTA design can handle the same amount of units in six Spartan x3c50 and can be easily scale by adding more FPGAs.

7. Conclusions and Future Work

A digital scalable hardware design of a Kohonen map has been presented. Though very simple when compared with the original algorithm, the proposed model succeeds in retaining the main features of the network, allowing data clustering with topology preservation.

To achieve scalability, the LWTA design "distributes" the Winner -Take- All block by spreading the "global" information locally, without the need of a central control unit, making plausible the interconnection of several FPGAs, each one configure with a part of the map, to implement a large map that otherwise would be impossible to implement in a single chip.
Consequently, in the LWTA design, each neuron is connected only with the neighbor neurons and all the neurons work independently.

The GWTA design has been used successfully on image segmentation problem and the LWTA design is taken into consideration for the image segmentation as well as perception of mobile robot.

8. References

1. M. Hassoun, "Fundamental of artificial neural networks" , MIT Press / Bradford Book, 1995.
2. T. Kohonen, "Self-Organization and Associative Memory", Springer-Verlag, Berlin, 1989.
3. Y. Liao, "Neural Networks in Hardware: A Survey", Department of Computer Science, University of California.
4. R. Newcom, J. Lohn. "Analog VLSI for Neural Networks", MIT Press / Bradford Book, 1995.
5. T. Schönauer, A. *et. Al*, "Digital Neurohardware", Technical University of Berlin, Berlin.
6. M. Skrbek, "Neural Networks-Hardware Implementation", Department of Computer Science and Engineerig, FEE CTU, Prague
7. D. Macq *et. Al*, "Analog Implementation of a Kohonen Map with On-Chip Learning", IEEE Transactions on Neural Networks, Vol 4, No 3, May 1993.
8. D. Ghosh , A. P. Shivaprasad, "Possibilistic Clustering in Kohonen Networks for Vector Quantization", Institute of Science, Bangalore, India.
9. S. Ruping *et. al.* "Hardware Design for Self Organizing Feature Maps with Binary Input Vectors", Lecture notes in Computer Science, Springer Verlag, pp. 488-493, 1993.
10. A. Upegui, E. Sanchez, "Evolving Hardware by Dynamically Reconfiguring Xilinx FPGAs". International Conference on Evolvable Systems, 2005.
11. J. Tian, J. Suontausta, "Scalable Neural Network Based Language Identification from Written Text", Nokia Research Center, Tampere, Finland.
12. A. E. Valencia A., J. A. Peña J. and M. Vanegas, "Digital Hardware Design of a One-Dimensional Kohonen's SOFM with Chessboard Norm and Exponential Neighboring Function", International Congress on Computational Intelligence, Monteria, Colombia, 2005.
13. Mentor Graphics, "ModelSim", in http://www.model.com.
14. S. M. Trimberger, "Field-Programmable Gate Array Technology", Kluwer Academic Publishers, 1994.
15. Xilinx Corp, "Xilinx: The Programmable Logic Company", in http://www.xilinx.com.