HARDWARE IMPLEMENTATION OF AN ARTIFICIAL NEURAL NETWORK WITH AN EMBEDDED MICROPROCESSOR IN A FPGA

Gisnara Rodrigues Hoelzle[2] and Fernando Morgado Dias [1,2]

[1] *Centro de Ciências Matemáticas – CCM, Universidade da Madeira, Campus Universitário da Penteada, 9000-390 Funchal, Madeira, Portugal*

[2] *Departamento de Matemática e Engenharias, Universidade da Madeira, Campus da Penteada, 9000-390 Funchal, Madeira, Portugal*

brasergis@hotmail.com, morgado@uma.pt

Abstract

This article describes the implementation in hardware of an Artificial Neural Network with an embedded Microprocessor in a FPGA. The implementation of a Neural Network in hardware can be desired to benefit from its distributed processing capacity or to avoid using a personal computer attached to each implementation. The relevance of implementing it in a FPGA comes from its flexibility, low power consumption and higher performance. This implementation uses an embedded processor. Embedding the processor allows achieving the benefits from hardware and from software in a single platform. The implementation with a microprocessor can be quite easy while a regular hardware implementation can be hard to develop.

The hardware implementation is based in a Feedforward Neural Network, with a hyperbolic tangent as activation function, with floating point notation of single precision. The device used was an FPGA Virtex II Pro XC2VP30, Xilinx with a MicroBlaze soft core processor. The microprocessor soft core subsystem occupied 1766 slices and it used 89KB of RAM for the application, data and results storage (the base value of on-chip memory is of 64KB). The Matlab was used to validate the implementation by comparing it with the results of the hardware solution while using data from a real system. The results show that the implementation does not introduce a noticeable loss of precision but is slower than the Matlab implementation running in a PC with a processor running at 2,8GHz.

Keywords: Artificial Neural Network, MicroBlaze, Hardware Implementation, Hyperbolic Tangent, FPGA, Embedded Microprocessor.

1. Introduction

The majority of the authors have shown that the solutions with ANNs (Artificial Neural Networks) reach better results in the implementation phase with specific hardware than the most common implementation using a personal computer or workstation [1]. The existence of these solutions in hardware is of extraordinary importance for areas as Applied Sciences and Health Sciences.

The ANNs are parallel distributed systems, since they have the capacity to receive several inputs at the same time and to distribute these inputs in an organized manner. With this architecture, the information stored and shared in all the processing units of the ANNs improve the performance and the reliability in the systems implemented [1].

The implementation of ANN will have to supply a signal to the output of the system that will activate the coupled components, resembling the one that happens in the Biological Neural Network.

The hardware implementation with the embedded processor needs to reproduce the architecture of the ANN needed. This means considering the number of inputs and outputs, the number of neurons, the number of connections to each neuron, the number of layers, the numerical representation and accuracy for the involved signs.

2. Implementation of the Neural Network

The implementation of Artificial Neural Networks has become a part of many scientific projects. They are specific projects and dedicated to the solution of complex problems that in principle would be insoluble.

2.1 *Hardware Platform*

The platform chosen for the implementation of the ANN was a FPGA with an embedded MicroBlaze. This solution was chosen because of its flexibility, greater performance and low consumption of energy.

In this project the platform used was a ML310 board, represented in figure 2.1, with a FPGA Xilinx Virtex-II Pro XC2VP30 FF896 (Flip-Chip Fine-Pitch BGA Package) speed grid -6, of the Xilinx. The MicroBlaze is the embedded microprocessor in the FPGA.

The MicroBlaze is a microprocessor with an RISC architecture (Reduced Instruction-Set Computer) of 32 bits. This microprocessor is soft core, i.e., it has a set of features configured through a Hardware Description Language (HDL). The MicroBlaze is optimized for implementations in FPGAs, a flexible processor system that's easy-to-use, area-efficient, optimized for cost-sensitive designs, and avoids processor obsolescence [2]. The number of MicroBlazes that can be embedded depends exclusively on the capacity of the FPGA.

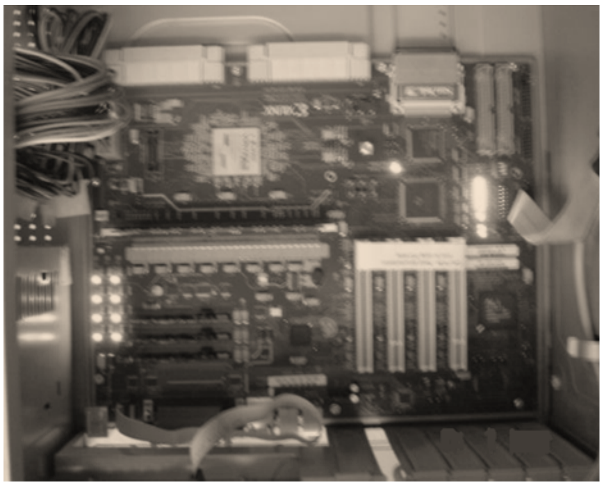


Figure 2.1 – Platform ML310

Figure 2.2 shows a functional block diagram of the MicroBlaze. The blocks in blank are the basic feature for the functioning of the system of the MicroBlaze and the blocks filled are optional.

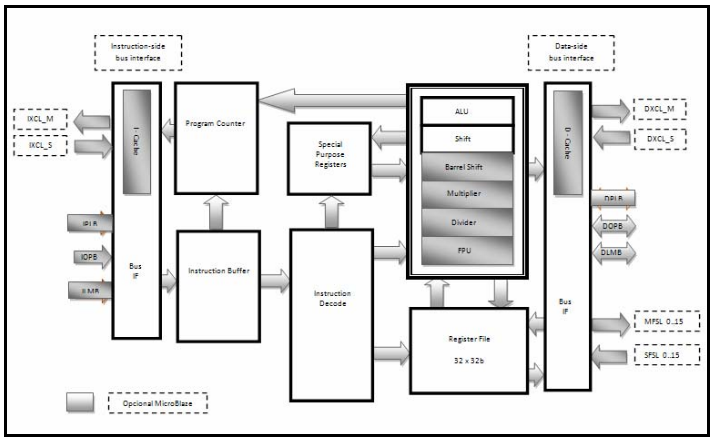


Figure 2.2 – MicroBlaze Processor Block Diagram [2]

2.2 *Implementation*

The type of Neural Network used in this project was Feedforward Neural Network, i.e. a network that allows only connections in the output direction. This option result of its bigger spreading at the level of the developed tools that allow the fast implementation of models [1].

The code programmed for the Neural Network was developed with the objective of being generic, i.e., the network would have to receive a variable number of inputs and neurons. The number of neurons of the hidden layer allows us to adjust the size of the network to the complexity of the system being used [1].

During the project, several architectures were tested. The notation used defines the architecture in a simple form: they quantify the number of inputs, neurons in the hidden layer and neurons in the output layer, i.e., for the network 3-8-1, we have 3 inputs, 8 neurons in the hidden layer and 1 neuron in the output layer.

For the internal calculations of the activation function, an hyperbolic tangent function was used (expression 2.1). This function supplies non linearity to the Neural Network.

However, in order to make the processing of the network in the hardware faster, the function was used in a reduced form, i.e., the function was implemented as stated in expression 2.2. The advantage is obvious since it has only one exponential function. In the output layer, the linear function of expression 2.3 was used.

In the expressions below, ***y*** represents the output and ***x*** represents the input.

$$\boxed{Y(x) \;=\; \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}} \qquad (2.1)$$

$$\boxed{Y(x) = 1 - \frac{2}{1+e^{(2x)}}} \qquad (2.2)$$

$$\boxed{Y(x) = x} \qquad (2.3)$$

For the internal representation floating-point notation with simple precision was chosen, since this allows the representation of larger number with the same number of bits, when compared to fixed point notation. To increase the processing speed of the hardware implementation, a Floating Point Unit (FPU) was added to the configuration of the MicroBlaze embedded system.

3. Tests and Results

Several tests were performed in the ANN implementation. The tests were used to verify the correct operation of the network, modifying the size of the system and comparing the results with the ones obtained with an implementation in Matlab using a toolbox developed by Magnus Noorgard [3].

The structure of two of the models used is represented in figures 3.1 and 3.2.

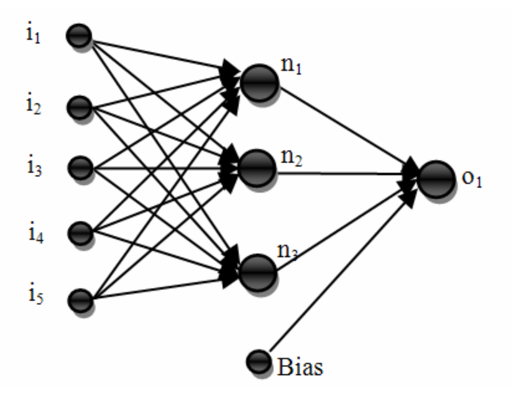


Figure 3.1 - Neural Network Model For6700

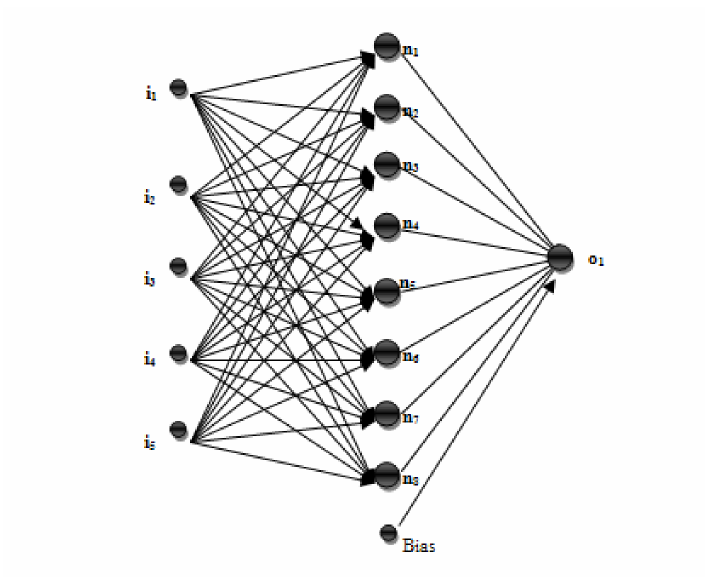


Figure 3.2 - Neural Network Model FORGA001

The embedded microprocessor was programmed in C language. ANNs of different size were implemented and memory occupation was between 90 and 100KB. This allows us to conclude that large networks can be implemented with the proposed solution.

After verifying the good operation of the network and test some different network sizes, the implemented solution was submitted to a comparison with an application developed in Matlab.

All tests carried through in Matlab were done using data from a real system. This system is an electric kiln that was built as part of a research project [4] and had the objective of developing a kiln with high degree of control of the profiles temperature for the ceramic industry and glass. The area of intended operation was around 750° C, there is an upper limit of 1200° C.

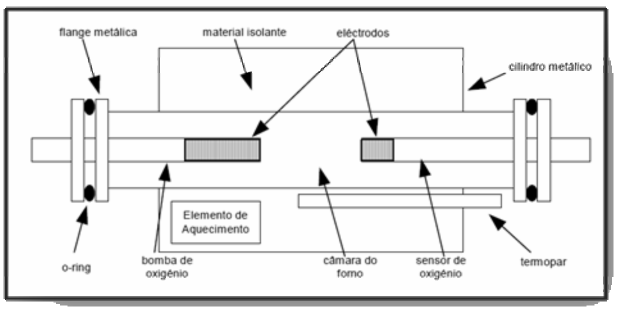


Figure 3.3 – Schematic view of the kiln [4]

4. Results Achieved

Three different reference files for the ANNs were used for the tests (ID050300, ID13100A and DT030103).

Figures 4.1, 4.2 and 4.3 show outputs obtained for reference files with ANNs models.

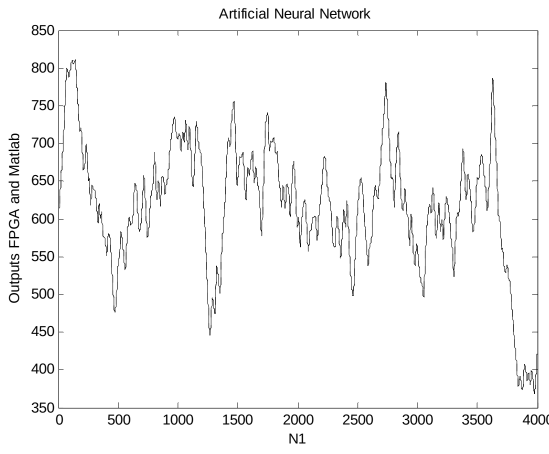


Figure 4.1 – Outputs obtained for reference ID050300 with model For6700

For this test the mean square error (MSE) was calculated between the outputs of the ANN implemented in Matlab and in the FPGA, resulting in a value of 8.3660x10^{-20}.

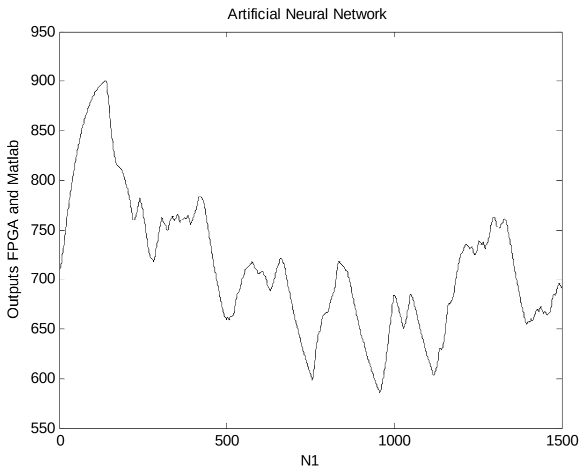


Figure 4.2 – Outputs obtained for reference ID13100A with model For6700

For this test the mean square error (MSE) was calculated between the outputs of the ANN implemented in Matlab and in the FPGA, resulting in a value of 8.5372x10^{-20}.

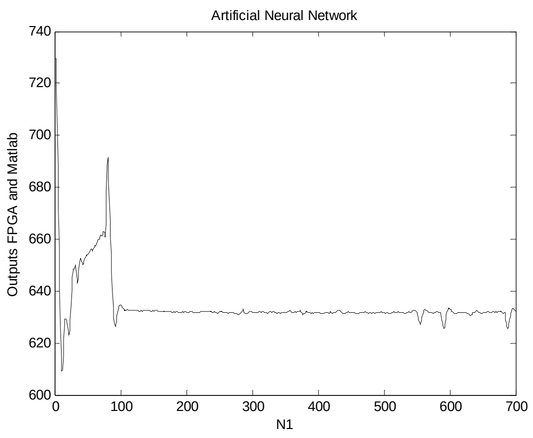


Figure 4.3 – Outputs obtained for reference DT030103 with model FORGA001

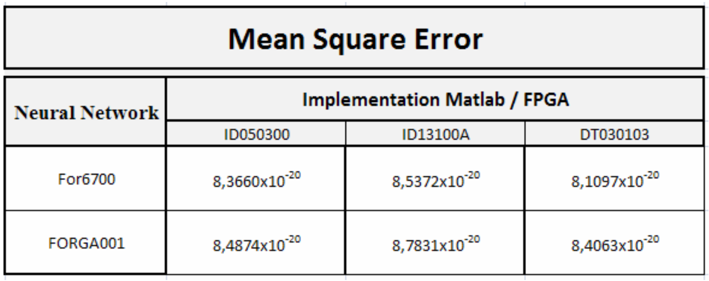
For this test the mean square error (MSE) was calculated between the outputs of the ANN implemented in Matlab and in the FPGA, resulting in a value of 8.4063x10^{-20}.

The MSE value was collected for all the tests and is presented in 4.1.

The analysis of the values allows us to conclude that the precision obtained in the FPGA implementation with the embedded MicroBlaze, is very good.

This behavior was expected since the networks have the same parameters and no approximation was used in the implementation.

Table 4.1 Mean Square Error of the ANNs Models



Mean Square Error

Neural Network	Implementation Matlab / FPGA		
	ID050300	ID13100A	DT030103
For6700	8,3660x10^{-20}	8,5372x10^{-20}	8,1097x10^{-20}
FORGA001	8,4874x10^{-20}	8,7831x10^{-20}	8,4063x10^{-20}

The implementation of the embedded MicroBlaze microprocessor in the FPGA Virtex-II Pro, developed in this work, used 12.89% of the available slices and 10.51% of the Look Up Tables (LUTs) with four inputs and with a maximum operating frequency of 120,465MHz.

In the hardware the time necessary from processing each of the tests was registered:

•For6700/DT030103 and FORGA001/DT030103 – 1434ms

•For6700/ID13100A and FORGA001/ID13100A – 2147ms

•For6700/ID050300 and FORGA001/ID050300 – over 42s

In these last tests an overflow occurred. Since the microprocessor was running at 100MHz and has 32 bits we can be sure the test took more than 42s.

In Matlab, using a 2,8 GHz processor, the following values of time, were obtained:

•For6700/DT030103 and FORGA001/DT030103 – 62ms

•For6700/ID13100A and FORGA001/ID13100A – 109ms

•For6700/ID050300 and FORGA001/ID050300 – 283ms

5. Conclusions

This work allowed verifying important aspects regarding the hardware/software implementation, including the flexibility and the time of development of project. The great potentiality of the hardware chosen for the implementation was also verified, observing that, after all the network configuration, we used only 33% of the resources offered by the FPGA.

The developed system was tested with different models of ANNs and using data from a real system. The tests made it possible to verify the performance of the network and allowed us to reach the conclusion the implementation is accurate.

Although this implementation is slower than the one made using a PC, a FPGA board is less expensive than a PC and more stable in the sense that it does not depend of an operating system.

Acknowledgment

The authors would like to acknowledge the Xilinx University program for support and software used in this work.

References

[1] Ferreira Pedro, Ribeiro Pedro, Antunes Ana, and Dias M. Fernando. A high bit resolution FPGA implementation of a FNN with a new algorithm for the activation function, Neurocomputing Vol. 71, Issues 1-3, Pages 71-77, December 2007.

[2] Xilinx: MicroBlaze Processor Reference Guide, available at http://www.xilinx.com/support/documentation/sw_manuals/mb_ref_guide.pdf, June of 2008.

[3] Magnus Noorgard. System Identification and Control with Neural Networks. PhD thesis, Department of Automation, Technical University of Denmark, 1996.

[4] Morgado Dias. Non-linear control techniques based on Neural Networks: from the algorithm to the hardware. PhD thesis in Electrical Engineering, University of Aveiro, 2005, available at http://dme.uma.pt/morgado/Down/TeseCompletaFMD.pdf, November 2008.

[5] Gisnara Rodrigues. Hardware Implementation of an Artificial Neural Network with an Embedded Microprocessor in a FPGA, to be presented at the end of September.