Journal of Theoretical and Applied Information Technology
28th February 2015. Vol.72 No.3
© 2005 - 2015 JATIT & LLS. All rights reserved·

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

COMPARISON OF HARDWARE AND NIOS II BASED-SOFTWARE IMPLEMENTATION OF MLP ON THE FPGA PLATEFORM

[1] BOUSSAA MOHAMED, [2]ATOUF ISSAM, [3]ATIBI MOHAMED, [4] BENNIS ABDELLATIF
[1, 2, 3, 4] Laboratory of information processing, Hassan II – Casablanca University
Cdt Driss El Harti, BP 7955 Sidi Othman Casablanca, 20702, Maroc
E-mail: [1] md.boussaa@gmail.com, [2] issamatouf@yahoo.fr , [3]atibi.simo@gmail.com, [4] al_bennis@yahoo.fr

ABSTRACT

The artificial neural networks provide optimal and satisfactory solutions for most applications of image and signal processing requiring complex mathematical operations. This often leads to implement these artificial neural networks in platforms that retain their performance (computational efficiency, speed of execution). This article presents, in a first stage, two implementation methods of artificial neuron network MLP types: hardware and software based on the processor Nios II and, secondly, comparing their performance in terms of execution time and material resources consumed. The two implementations have been done in the FPGA platform DE2-70 Altera that offers performance in terms of flexibility, efficiency and very remarkable speed.

Keywords: *MLP, NIOS II, FPGA, hardware resource, Execution time.*

1. INTRODUCTION

The artificial neural networks have become solutions for a wide variety of applications in various fields such as industrial engineering, robotics, automotive etc...

ANN is a network, whose neurons are interconnected in architecture, and links are configured through a learning process for a specific application, like pattern recognition or classification; however, as in all biological nervous systems, learning process requires adjusting the connections (synaptic weights) between neurons. The importance of these decision systems is increasing as the data are available in several application areas. [1][2][3].

Among the most architectures used in the literature, there is often the architecture multilayer perceptron (MLP). It is a layered architecture in which synaptic weights are adjusted by the algorithm of Back propagation (BP) in the learning phase. [4]

The major problem frequently encountered during the use of multilayer perceptron is related to their implementation in embedded systems, which are capable of maintaining the performance of these neurons. Thus, several researchers have tried to implement this system in embedded platforms, most used in the literature is the Field Programmable Gate Array (FPGA). [5]. Despite some attempts that have been made in this direction, it was found that there is always a trade-off between the hardware implementation of neural network and computational efficiency (the implementation of the activation function without use of approximations [2] [18]).

The FPGA is a programmable digital hardware that offers many new perspectives for applications requiring the MLP, as the possibility of optimizing the hardware and parallel processing. Furthermore, the design made for a specific component of FPGA can be easily installed in another having similar capabilities, and can be changed as required. [6]

The growth in the size and performance of artificial neural networks multilayer perceptron pushes designers to use soft processors for controlling systems using a large number of blocks that remain intellectual property (IP). [6]. this gives rise to the creation of soft processors designed by the developers of FPGAs.

A soft processor is a heart microprocessor that can be fully implemented using logic synthesis. It can be implemented through the different semiconductor device containing programmable logic (e.g. FPGAs, CPLDs).There are several soft cores available on the market, for example, Pico Blaze, Micro Blaze, Open fire, Nios, Nios II, Cortex-M1, Mico8, and Mico32. [7].The NIOS II is a 32-bit soft processor with a simple structure that can be integrated into FPGAs.

This document aims, as a first step, to describe the stages of implementation of artificial neural networks in the FPGA platform, without using the approximation affecting the efficiency of calculation, according to two methods, hardware implementation and software implementation, using a soft microprocessor NIOS II, and, as a second step, to compare the execution time and physical resources of these two methods.

2. THEORETICAL STUDY

2.1 Artificial neuron networks

The artificial neural networks are networks that modeling the network architecture of natural neurons. They consist of two main parts, the neuron model (base unit) and the connection model between neurons [8] [9].

There are several models of neuron, the most used in the literature and applications are based on the model developed by McCulloch and Pitt's model (figure 1). This model receives at its input two vectors; the parameter vector dedicated to the application performed and the other connection vector of weights named synaptic weight, for it performs a weighted summation between the two vectors (1).The result of this operation is transferred to the output through a transfer function (activation function).The values of these synaptic weights are already set during the learning phase.

$$V = \sum_{i=0}^{n} Wi.Xi \qquad (1)$$

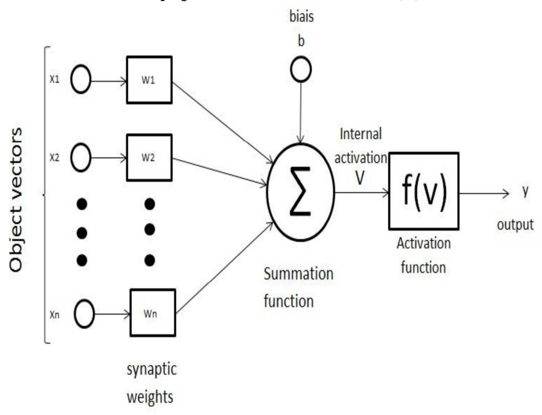


Figure1. Formal neuron model

With:

X1……Xn: vector object of the neuron.

W1……Wn: synaptic weights in the neuron.

∑: a function that calculates the sum of the multiplication between the object vector and the synaptic weights according to equation (1).

B: The bias of the summation function.

F (V): Activation functions of the neuron.

Y: Output of the formal neuron.

The model of the formal neuron used as an activation function is the sigmoid function (figure 2), which has several advantages compared to other functions such as the threshold function and Gaussian. There may be mentioned among its benefits derivability and continuity of function.

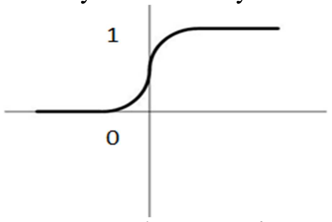


Figure2. Sigmoid activation function

The topology of an artificial neuron network is defined by the choice of architecture and the choice of adjustment algorithm synaptic weights.

Most artificial neural networks use multilayer perceptron architecture (figure 3); this architecture uses in addition to the input layer and the output layer an intermediate layer called hidden layer (one or more).The network architecture is described by the number of layers and the number of neurons in each layer.

Most networks type multilayer perceptron use an adequate algorithm for the adjustment of the synaptic weights. Among these algorithms, and the most used, is the algorithm of back propagation of errors (Back pro) that simulates the phenomenon of learning by error. The objective of this algorithm is to minimize the overall error in using the method of gradient descent. The latter consists of randomly initializing the values of the synaptic weights, and then successively introducing the elements of the training set to the network input and evaluating, for each of the base member of example, the error observed at the output of each neuron, to exploit the value of this error to adjust the value of the synaptic weights.

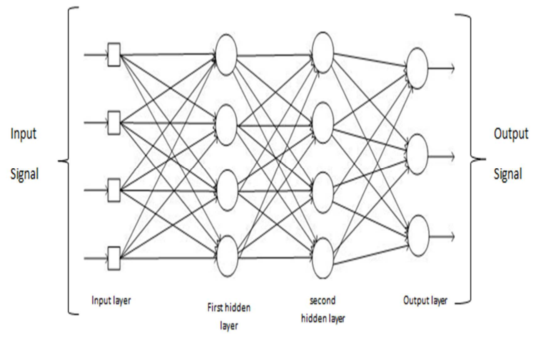


Figure3. Architecture of the multilayer perceptron

2.2 FPGA

The Field Programmable Gate Array is reprogrammable integrated circuits. They offer the ability to make digital more or less complex functions with significant benefits or parallel processing [10], whether a hardware offering speed, flexibility and low cost flexibility [11][6], to achieve the desired numerical functions. The advantage is that the same chip can be used in many different electronic systems.

This is a type of integrated circuit (IC) containing an array of logic cells that can be programmed by a user to act as an integrated circuit arbitrary (figure 4) [12]. For example, one can implement a transformer, a digital filter or an interrupt controller on the FPGA. The largest FPGAs allow the user to create complex systems on chip that contains several interconnected components. The FPGAs used as the programming language hardware description language VHSIC Hardware Description Language (VHDL) or Verilog. The FPGA configuration data (the micro software) are then created by specific software that description.

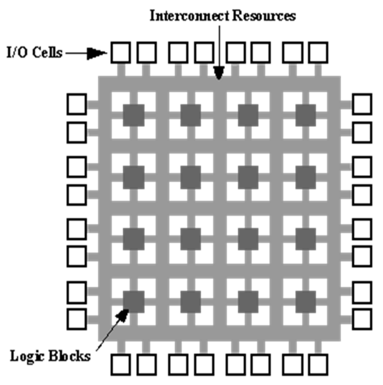


Figure4. Logic cell array architecture of FPGA

The FPGA includes some modules: memory for general use, the "DSP" cabled blocks, the phase locked loop for clock generation.

Commercially, FPGAs have taken a place in the market for embedded systems; this is due to the low cost of developing applications on this system, and the flexibility of programming and reprogramming several times [13].

2.3 NIOS II

The development, the complexity and reliability of current FPGAs has given birth to a very close discipline of System on Chip SOC called the System on Programmable Chip (SOPC). The SOPC is to design a system On Chip either from existing IP (Intellectual Property) or from a description of A to Z language HDL generally with VHDL or Verilog standards and implementing them on FPGA. The basis of any modern digital system is the use of one or more microprocessors. In the field of the SOSH, these microprocessors are descriptions in the language HDL (VHDL or Verilog) or compilations of these descriptions named "Netlists" from which the name "SOFTCORE". [14]

These "softcore" microprocessors have the advantage of being more flexible with respect to "hardcore" microprocessors which are in the form of integrated circuits.

The NIOS II soft-core is a 32-bit microprocessor, developed by Altera (figure 5). The NIOS were introduced in 2001 to be the first marketed and used processors in the industry. Its creation was specifically for the design of embedded systems in FPGA. Its compact and advantageous architecture uses less hardware resources FPGA comparable to the 8-bit microprocessors. It is designed specifically for the Spartan-3 and Virtex cyclone. [15][7].

The NIOS II microprocessor has the following characteristics:

- 32-bit microprocessor RISC.
- HARVARD internal architecture.
- 32 general purpose registers and six special purpose registers.
- Machine instructions are also 32 bits.
- 6 levels of maximum pipeline.
- 30 to 80 MIPS (Million Instructions per Second).
- Frequency up to 200 MHz

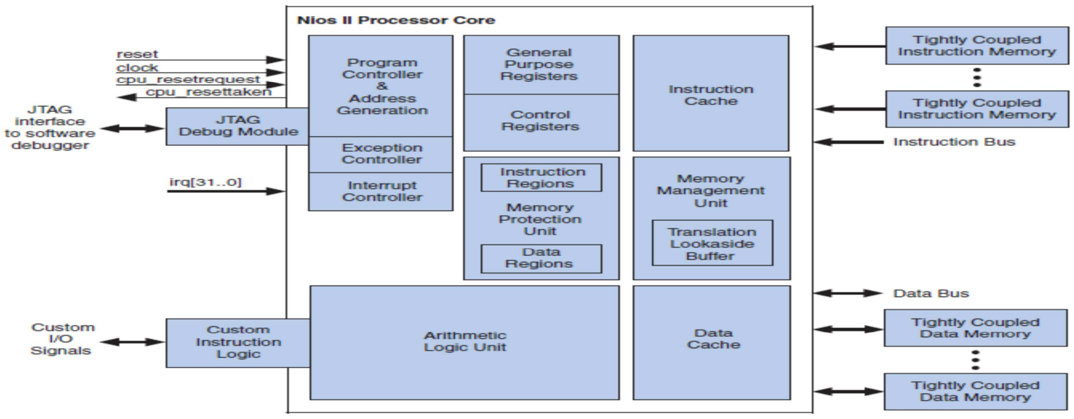


Figure5. NIOS II processor

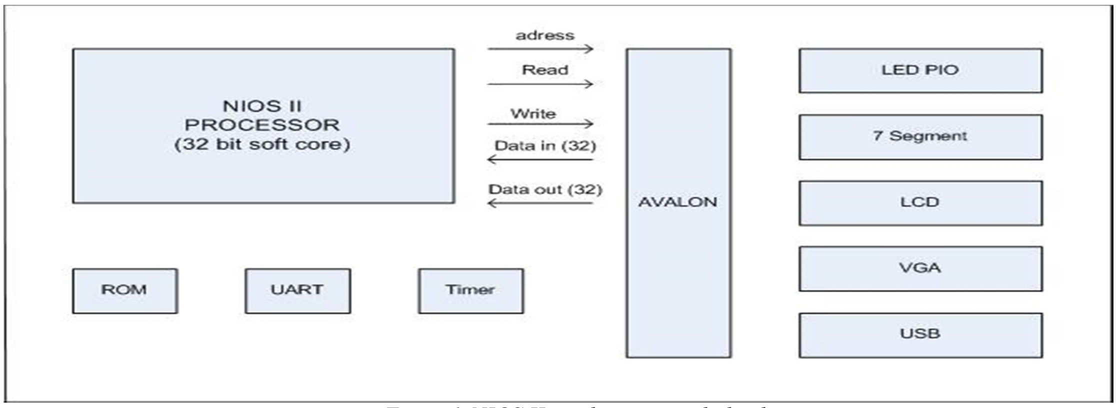


Figure6. NIOS II combination with the devices

- Ability to integrate a hardware multiplier and divider.

The combination of the microprocessor and a set NIOS II device on the same chip is equivalent to a microcontroller (figure 6). [6].

3. IMPLEMENTATION DETAIL

This section describes the different steps and modules necessary to implement the artificial neuron network multilayer perceptron type from the FPGA platform by two implementations: Implementing hardware and software implementation based on the use of the microprocessor soft-core NIOS II.

3.1 Hardware implementation

The hardware implementation of the MLP network [17] (figure 7), described in VHDL hardware description language, and consists in implementing a number of formal neuron interconnected between them according to a well-defined architecture. Each formal neuron performs a series of more or less complex mathematical operations.

The proposed modular implementation based on the use of formal neuron in the form of a module that includes all the operations performed by him and a number of Blocks that provide synchronization and network operation.

The idea of this implementation is to replace the set of formal neurons that constitute a layer (hidden layer or output) by one module of the neuron that performs the work of all the neurons in this layer.

3.1.1 Module of the formal neuron

The neuron is the base element of artificial neuron networks. This element is divided into two parts. In the first part, the neuron carries out a number of mathematical operations (multiplication and addition) to calculate its own internal activation (1) and in the second part, the neuron computes its output through a transfer function (sigmoid).

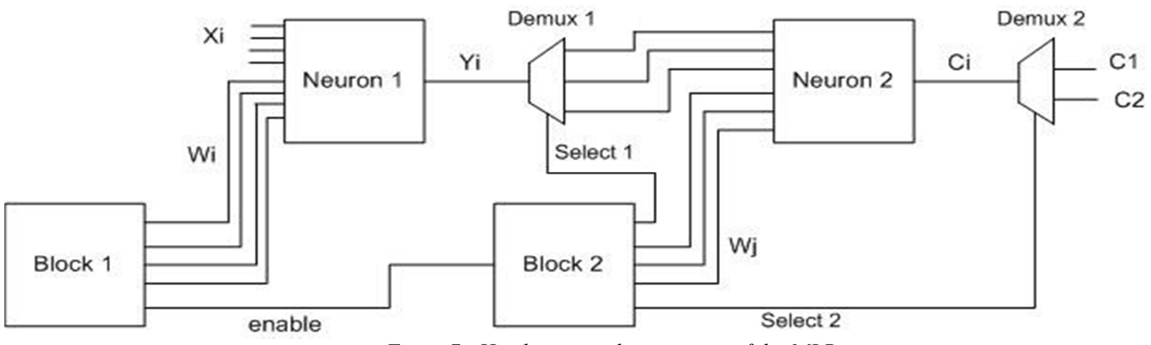


Figure7. Hardware implementation of the MLP

The implementation of these two parts (sum, multiplication, division and exponential) requires the use of mega functions blocks described in hardware description language (VHDL), which showed an accuracy of calculation of interest in the design. These mega functions blocks are the intellectual property (IP) proposed by the developers of simulation software QUARTUS.

The combination of these two parts (internal activation part and the transfer function part) is used to represent the formal neuron as a module according to the following scheme (figure 8):

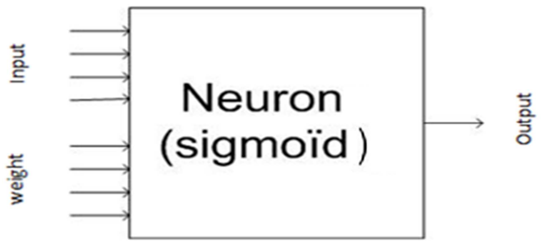


Figure8. The formal neuron implementation

In the proposed (figure 7) architecture, each module of the formal neuron represents all the neurons in a layer of the layers of the MLP. The functioning of this module is to calculate the first output of the first neuron in that layer according to its two input vectors (weight synaptic and vector input which can be either parameters input or output neurons of the previous layer), and the second time to compute the output of the second neuron in the same layer. This process is repeated until the last neuron of this layer.

3.1.2 Blocks module 1 and 2

In this architecture, the module of formal neuron needs to receive a series of vectors in which each vector is the vector of the synaptic weights of a neuron among the neurons of a layer of the MLP. This is ensured by the use of blocks 1 and 2 (figure 9a.9b).

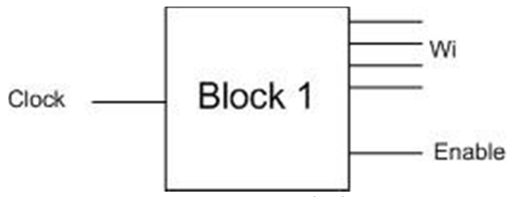


Figure9a. Block 1

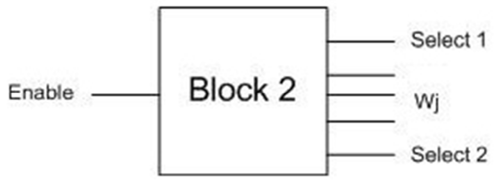


Figure9b. Block 2

The block 1 generates a series manner and synchronized synaptic weight vectors (vector Wi) which are previously stored in the memory and Control the validation of block 2 (enable).

After its validation, the block 2 generates in the same way synaptic weight vectors (vector Wj) and controls the path of flow of the output vector (Yi) of the neurons of the preceding layer via the select control signal 1 and the output vector (Ci) of the neuron output layer 2 via the select signal.

3.2 Software implementation

The implementations of the MLP network in the FPGA platform based on the NIOS II improves efficiency and reduces design complexity on the FPGA platform; this implementation requires two essential steps as shown in the following diagram (figure 10).

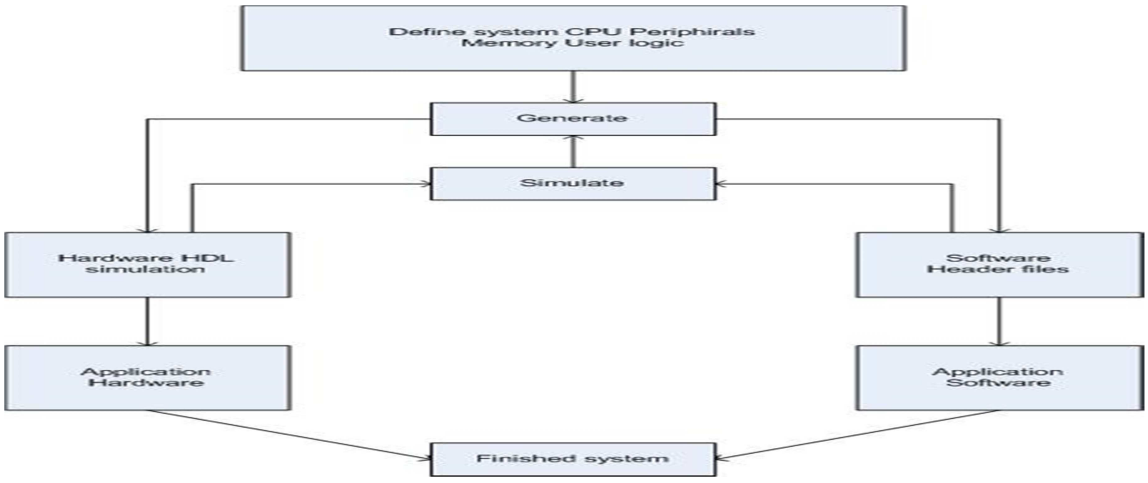
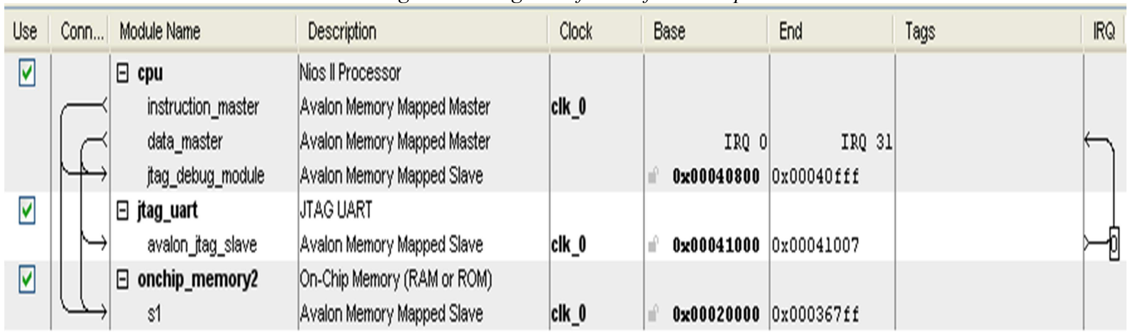


Figure10. Diagram of the software implementation



Use	Conn...	Module Name	Description	Clock	Base	End	Tags	IRQ
✓		cpu	Nios II Processor					
		instruction_master	Avalon Memory Mapped Master	clk_0				
		data_master	Avalon Memory Mapped Master		IRQ 0	IRQ 31		
		jtag_debug_module	Avalon Memory Mapped Slave		0x00040800	0x00040fff		
✓		jtag_uart	JTAG UART					
		avalon_jtag_slave	Avalon Memory Mapped Slave	clk_0	0x00041000	0x00041007		0
✓		onchip_memory2	On-Chip Memory (RAM or ROM)					
		s1	Avalon Memory Mapped Slave	clk_0	0x00020000	0x000367ff		

Figure11. The material generated by SOPC Builder

3.2.1 Hardware part

This section is devoted to the implementation of system-on-chip based microprocessor NIOS II; the hardware implementation of NIOS II requires 2 development tools QUARTUS and SOPC Builder.

The Quartus II of Altera design software provides a complete design environment and platform that easily adapts to the specific needs of design. This is a complete environment for designing SOPCs. The Quartus II software includes solutions for all phases of FPGA and CPLD design. [7].

The SoPC is a new concept and an approach proposed by Altera.[16] It has features such as programmable logic elements combined EDA, SOC, DSP and IP, which help the flexible design, the ability to customize the application, expansion, scalability and software programming and hardware development system. The evolution of design technology SoPCs is the product of several factors: the modern technology of computer-aided design, the EDA technology and the huge development of the technology of large-scale integrated circuits. [11].

As shown in the figure 11, the SOPC builder can simulate and generate all the necessary material for a given implementation as: NIOS II microprocessor, memory (ROM, RAM, SDRAM, DRAM and FLASH), PIO LED, LCD, and Avalon bus...After the complete compilation and generation, the SOPC builder loads the result as a single module (figure 12) to be added to the component library of the FPGA. This module (NIOS II and its peripherals) is the equivalent of a simple microcontroller which has its own set of instructions and can be programmed with evolved languages like C for example.

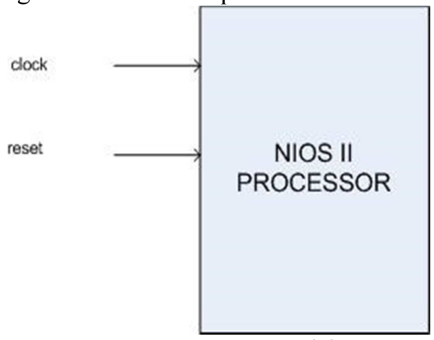


Figure12. NIOS II module

3.2.2 Software part

The programming tool generated module is the NIOS IDE; it is based on Eclipse to which Altera has added plug-ins and of course a compiler and an assembler for the NIOS II processor. NIOS IDE allows programming with three languages which are C, C ++ and the assembler. The Nios II EDS provides two separate streams of development and includes many proprietary and open-source tools for creating Nios II programs. This platform is adapted to the implementation of software projects integers, since all development, debugging and state diagrams, can be made in one window. Eclipse IDE also provides different software models or error messages and functions as a terminal when executing code on a Nios II processor. The terminal displays any printed message using the C library functions such as printf (). [11].

The following figure 13 shows the diagram of a data application.

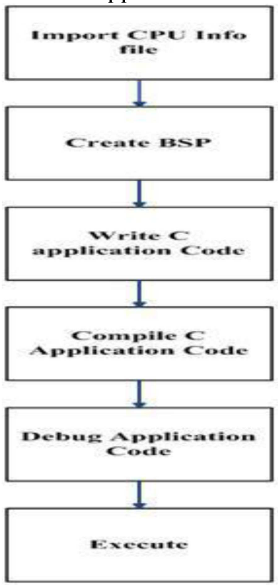


Figure13. The diagram of a data application

After building the final system (hardware part and software part), the implementation of the MLP or any application requires the input of the MLP code in C language described in the Eclipse editor, then compile the code and then load it into the hardware part that handles the run (figure 14).

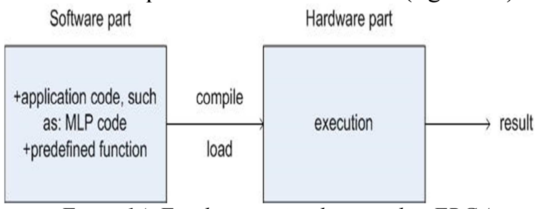


Figure14. Final system implemented in FPGA

4. RESULTS AND DISCUSSION

In this section, the two hardware implementations and software of the artificial neuron network multilayer perceptron that were proposed in the previous section, will be simulated and tested in FPGA Version DE2_70 EP2C70F896C6.This section will be devoted to levy execution time and hardware resources used by the two implementations in order to compare the performance of these two implementations.

To test the two implementations (hardware and software), the architecture used is 4-3-2 whose input consists of 4 layer neurons, the hidden layer composed of three neurons and the output layer composed of two neurons.

This architecture receives the same vector of the synaptic weights and the same input vector for the two implementations.

The result of the first test (hardware) of the artificial neuron network is performed with the option WAVEFORM VECTOR of the QUARTUS II 9.1 software. While the result of the second test (NIOS II) is performed with the software Nios II Software Build Tools for Eclipse. Both results are shown in the following table:

Table1. Results of two implementations

		software Architecture	hardware Architecture
Exécution time		1.0 s	1.9 us
Surface in the FPGA platform	Registre	1772 (3%)	9887 (14%)
	Combunational functions	2801 (4%)	13889 (20%)
	Logic element	3099 (5%)	15438 (23%)

The results obtained in the two implementations (table1) allow for a comparison, on the one hand, in terms of material resources used for each implementation and , on the other hand, in terms of execution time. Regarding the hardware resources, the software architecture implemented by the NIOS II remains the most optimized architecture with respect to the hardware architecture; this is an implementation that requires less hardware resource; this is due to the use of optimized developers Altera processor.

By cons, for the execution time, the hardware architecture offers undeniable advantages in terms of computational speed compared to software architecture (Nios II) that requires a period of time 106 times longer.

The two approaches have advantages one with respect to the other during the implementation.

The hardware architecture has a remarkable execution speed even with architectures that use enough space available in the platform used; however, the software architecture uses less hardware resource with same architecture that has a large capacity (figure 15).

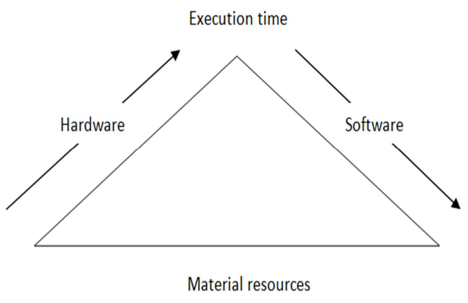


Figure15. Final scheme of comparison between the two implementations

5. CONCLUSION

The objective of this article has been initially to implement the artificial neuron network MLP type in the FPGA platform DE2_70 Version EP2C70F896C6 by two different methods (hardware and software based on Nios II), and, secondly, to compare the performance of these two implementations in terms of execution time and hardware resources used. This comparison has highlighted the performance of the hardware implementation before the implementation software with respect to the execution time and its poor performance in relation to the material used.

Both hardware and software implementations will make the FPGA platform very solicited in many real-world applications which need using the artificial neuron networks.

REFRENCES:

[1]-EDDINE, K. S., ABDELLATIF, H., ISSAM, A., & MOHAMMED, M. (2013). Serial Hardware Implementation of the MFCC and MLP Architecture on FPGA Circuit. International Journal of Engineering & Technology (0975-4024), 5(4).

[2] Savich, A. W., Moussa, M., & Areibi, S. (2007). The impact of arithmetic representation on implementing MLP-BP on FPGAs: A study. Neural Networks, IEEE Transactions on, 18(1), 240-252.

[3] M. Karthikeyan, Mr. Radhakrishnan. Cost Effective PC Based Oscilloscope using NIOS 2. International Journal of Research in Science & Technology (IJRST). ISSN: 2349-0845, Volume-1, Issue-4, May 2014.

[4] Rumelhart D.E, HINTON, G.E & WILLIAM, R.J.(1986). Learning representations by back-propagation errors. Nature, lond. 323, 533-536.

[5] Zhang, D., Wang, Q., & Xu, C. (2013). DATA ACQUISITION SYSTEM OF A SELF-BALANCING ROBOT BASED ON FPGA. Journal of Theoretical & Applied Information Technology, 47(2).

[6] Moslehpour, S., Jenab, K., & Valiveti, S. (2012). GPS Time Reception Using Altera SOPC Builder and Nios II: Application in Train Positioning. International Journal of Industrial Engineering, 23(1), 13-21.

[7] Pokale, M. S. M., Kulkarni, M. K., & Rode, S. V. (2012). NIOS II Processor Implementation In FPGA: An Application of Data Logging System. International Journal of Scientific & Technology Research, 1(11).

[8] ATIBI, M., BENNIS, A., & BOUSSAA, M. (2014). PRECISE CALCULATION UNIT BASED ON A HARDWARE IMPLEMENTATION OF A FORMAL NEURON IN A FPGA PLATFORM. International Journal of Advances in Engineering & Technology, 7(3).

[9] Boussaa M, Bennis A, Atibi M. (2014) Comparison between Two Hardware Implementations of a Formal Neuron on FPGA Platform. International Journal of Innovative Technology and Exploring Engineering, Volume-4 Issue 1.

[10] Karthikeyani, V., & Begum, I. P. (2013). Comparison a Performance of Data Mining Algorithms (CPDMA) in Prediction Of Diabetes Disease. International Journal on Computer Science and Engineering, 5(3).

[11] Moslehpour, S., Jenab, K., & Pabla, B. S. (2012). Implementing a soft core NIOS II processor for VGA application. International Journal of Engineering Research and Innovation, 4(2), 12-26.

[12] Shnidman, N. R., Mangione-Smith, W. H., & Potkonjak, M. (1998). On-line fault detection for bus-based field programmable gate arrays. Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, 6(4), 656-666.

[13] SATHISH KUMAR K,BELWIN EDWARD J, SARAVANAN B, SUDHAKAR N, PRABHAKAR KARTHIKEYAN S, RAVI K. A NEW POWER SYSTEM RESTORATION AND RECONFIGURATION USING FIELD PROGRAMMABLE GATE ARRAY. Journal

of Theoretical and Applied Information Technology. 15 May 2012. Vol. 39 No.1

[14] http://www.altera.com

[15] Mukadam, M. S. B., & Titarmare, A. S. (2014). Design of Power Optimization using C2H Hardware Accelerator and NIOS II Processor.

[16] Wei, Z., & Jun, S. (September, 2011). A high-speed serial data acquisition scheme based on Nios II. International Conference on Electronics, Communications and Control (ICECC). 2417-2420.

[17] ATIBI, M., ATOUF, I., BOUSSAA, M., & BENNIS, A. Parallel and Mixed Hardware Implementation of Artificial Neuron Network on the FPGA Platform.

[18] Zhu, J., & Sutton, P. (2003). FPGA implementations of neural networks–a survey of a decade of progress. In Field Programmable Logic and Application(pp. 1062-1066). Springer Berlin Heidelberg.