Journal of Engineering and Development, Vol. 17, No.6, December 2013 , ISSN 1813- 7822

FPGA Implementation Of Multilayer Perceptron For Speech Recognition

Asst. Lecturer Ghassan Hazin Shakoory
Mechatronics Eng. Dept.
College of Engineering, University of Mosul
E-mail: gasan62ha@yahoo.com

Abstract:

In this paper, a method for designing and implementing of Multilayer Percepton (MLP) based on BP algorithm has been suggested. The method has described a MLP on Register Transfer Level (RTL) using VHDL description language and implemented on Field Programmable Gate Array (FPGA) for speech recognition. Firstly, a multiply-accumulate (MAC) unit, and sigmoid nonlinear function are implemented as a basic building units of the MLP. The MLP is trained by BP learning algorithm. The optimized parameters are obtained by Matlab simulation for off chip training design. The implementations have been developed and tested on Xilinx Spartan-IIIE XC3S500E FPGA chip for embedded systems using Xilinx ISE 10.1 software. The research also presents a summary of the performance cost and data throughput with regards to the speed and required computational resources. The proposed hardware architecture are found to be 6 times faster than the software implementation.

Keywords: Multi-Layer Perceptron (MLP), ANNs, FPGA,VHDL, Speech Recognition.

تنفيذ شبكة المدرك متعددة الطبقات على شريحة FPGA لأغراض تمييز الكلام

م.م. غسان حازم شكوري
قسم هندسة الميكاترونيكس
كلية الهندسة/ جامعة الموصل

الخلاصة :

في هذا البحث، تم اقتراح طريقة لتصميم وتنفيذ شبكة المدرك متعددة الطبقات (MLP) والمبنية على خوارزمية الانتشار العكسي للخطأ (BP). تم وصف شبكة MLP في هذه الطريقة بمستوى تصميم المسجلات (RTL) باستخدام لغة وصف الكيان المادي (VHDL)، وتم التنفيذ على شريحة مصفوفة البوابات المبرمجة حقليا (FPGA) لأغراض تمييز الكلام. صممت وحدة الضرب والتجميع (MAC) أولا، ثم دالة التفعيل الغير خطية نوع سكمويد كوحدات أساسية لتنفيذ البنية المادية الكاملة لشبكة MLP. أستخدم برنامج محاكاة في البيئة البرمجية Matlab لإيجاد القيم المثلى لشبكة MLP باستخدام خوارزمية BP . أن الهدف من هذا البحث هو تطوير واختبار التنفيذ على كيان التشكيل المطاوع لشريحة نوع Xilinx Spartan-IIIE XC3S500E FPGA ولأغراض التطبيق في الأجهزة المضمنة باستخدام برنامج Xilinx ISE 10.1 . قدم هذا البحث أيضا ملخص الاداء وعامل العطاء نسبة للسرعة ومصادر الشريحة المستخدمة. حيث وجدت معمارية الكيان المادي المقترحة أسرع 6 مرات مقارنة مع الكيان البرمجي.

1. Introduction

Artificial Neural Networks (ANNs) are currently used in a wide variety of applications either for function approximation and data fitting or classification and speech recognition. One of the most common architectures consists of Multilayer Perceptron (MLP) network trained by Backpropagation (BP) algorithm. This approach is possible for offline design process, for applications where training data is static, or where conditions initially determined will stay the same for the duration of network's useful function. However, when online training is necessary or when the solution space is dynamic and new data is being added continuously, there is a critical need for testing wide range of topologies in a limited time period [1].

General Purpose Processors (GPPs) and ASICs have traditionally been the common means for building and implementing Artificial Neural Networks (ANNs). However such computing paradigms suffer from the constant need of establishing a trade-off between flexibility and performance. Due to the technological advance in the development of programmable logic devices, Field Programmable Gate Array (FPGA) have become attractive for realizing ANNs. FPGAs have shown to exhibit excellent flexibility in terms of reprogramming the same hardware and at the same time achieving good performance by enabling parallel computation [2]. The implementations have been developed and tested into FPGA for speech recognition systems, such as portable real-time systems for voice controlled phone dial systems, toys, aids for the disabled, etc. Among the different ANN models used for speech recognition, we focused on hardware implementation of a MLP. Recently, different authors have implemented MLP models on FPGA devices with different methods depend on the applications [1-9]. Pandya et. al. [2] employed neuron using LUT sigmoid approximation to implement three partially parallel architectures and a fully parallel architecture to realize the BP on Virtex2000e FPGA using Handel-C HDL. . Rosselló et. al. [4,5] presented a novel architecture for implementation of ANN on FPGA. Sarvda Chauhan et. al. [6] used MLP-BP algorithm for pattern recognition and classification applications with 60% rate.

The paper is organized as follows. Section 2 introduces the MLP structure adapted for speech recognition applications. In Section 3, the implementation of MLP at RTL level are described using VHDL description language. The results are summarized in Section 4, followed by the conclusions in section 5.

2. Structure of MLP for Speech Recognition

Isolated word recognition task is the process that maps an acoustic speech signal to single words at a time [9] (e.g., voice dialing on cell phones with a small vocabulary). There are three basic steps to performing recognition. First, we digitize the speech that we want to recognize; for telephone speech the sampling rate is 8000 samples per second. Second, we compute features that represent the spectral-domain content of the speech (regions of strong energy at particular frequencies). These features are computed with a frame. Third, a multi-layer perceptron (MLP) neural network is used to classify a set of these features, the features send in a context window to a MLP for classification (24 features per frame at 10 frames = 240 features). The output of the MLP neural network is the recognition of phonemes, groups of phonemes, or words **Figure. (1).**

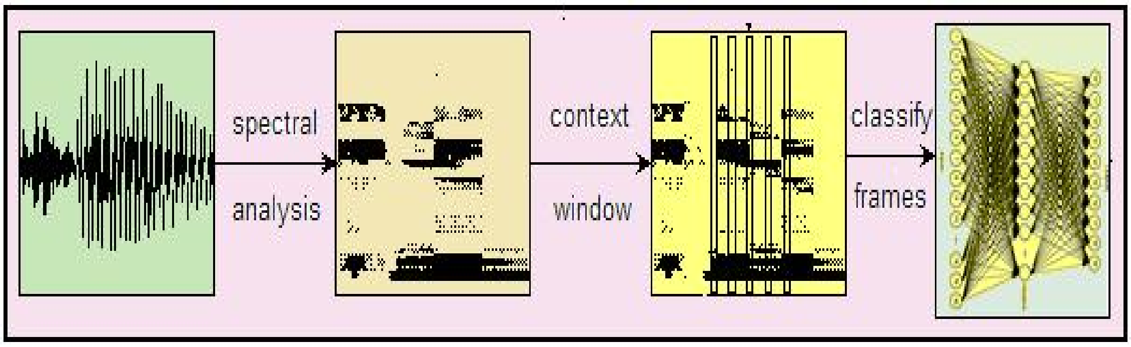

Fig. (1) Isolated word speech recognition process with MLP neural network[9].

The MLP model consists in a network of processing neurons arranged in layers. Typically, it requires three layers of neurons, an input layer which accepts the input variables used in the classification procedure, hidden layer, and an output layer with one neuron per class, **Figure. (2).** And **Figure. (3)** shows the Mean Square Error (MSE) curve of Performance of the learning algorithm train over 1000 epochs.

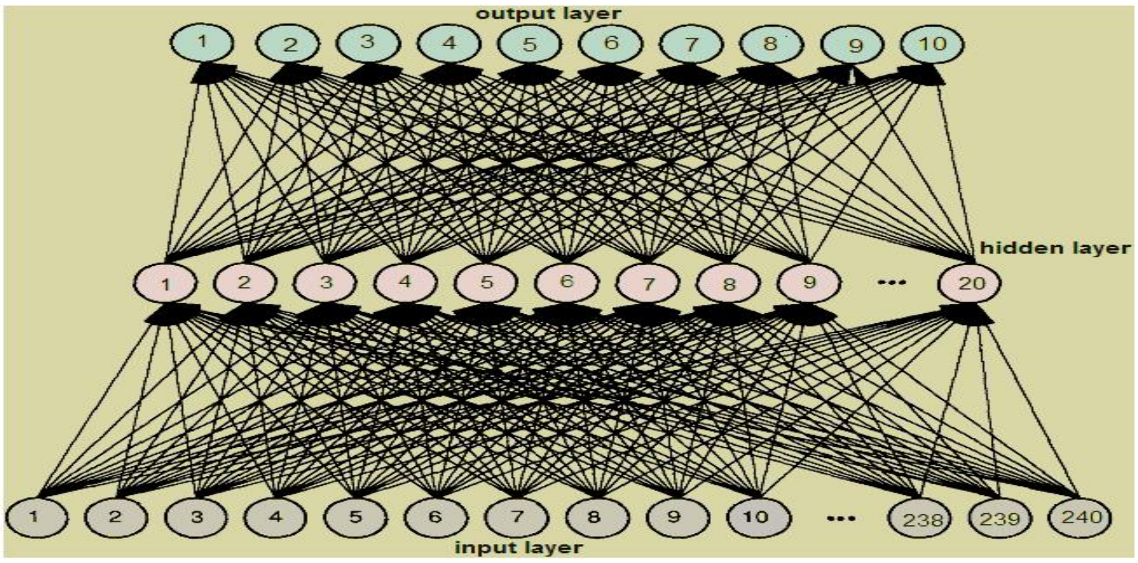

Fig. (2) The MLP for isolated word recognition.

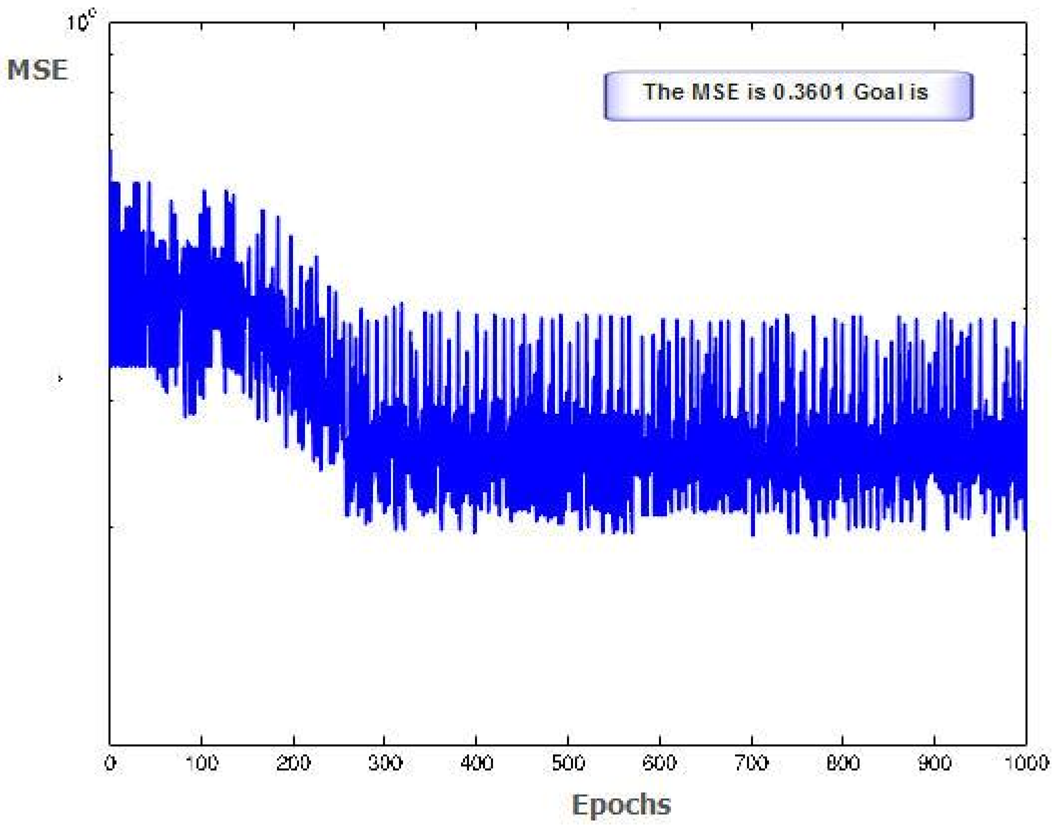

Fig. (3) Performance of the learning algorithm train over 1000 epochs.

3. VHDL Design and Implementations of the MLP

In practice, there are a lot of applications which require embedded processing in portable devices, and which are restricted by very constraining features, such as low cost, low power, and reduced physical size. We have implemented a MLP-based system in hardware as speaker-independent isolated word recognition platform. The MLP neural network in this application, has 240 input neurons, one for each feature in the 10-frame context window. There are 20 hidden neurons, and 10 output neurons in the output layer (corresponding to the 10 recognizable words).

For the MLP implementation, fixed-point (FXP) arithmetic is used for encoding the parameters and performing the computations. The choice for the number of bits to inputs, weights, and outputs of sigmoid function all must have the same range to easily manage multiple-layers processing. An 8 and 16 bits fixed-point number representation is chosen for this task. The inputs to the activation function is defined by a Look-Up Table (LUT) storing the useful values. The main storage strategy is to use RAM modules so that the inputs, the outputs, and the associated weights will be stored in these RAM modules. The RTL design of the MLP was implemented using standard VHDL and all the design processes have been carried out using ISE 10.1 software tool from Xilinx.

The most basic element of the neural network is the neuron, which multiply each input by its corresponding weight, and sum the multiplication. This operation is performed by multiply-accumulator (MAC) processing element, then followed by sigmoid function calculation. The structure of serial Processing Element (PE) with one MAC unit is shown in **Figure.(4).** Each input pair is multiplied together and a running total is recorded. The main advantage of this processing is the small constant area required, and the obvious disadvantage is the processing speed.

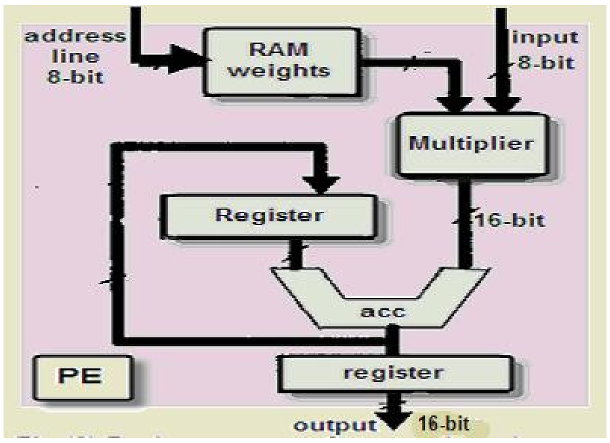

Fig. (4) Computing model structure of serial PE (processing element) of neuron.

The MAC process is further pipelined with a sigmoid function calculation process. If we denote || for a pipeline implementation, a rough depiction of computation at a single neuron in a layer in the feed-forward stage can be given by:

((Multiplication || Addition) || Sigmoid Calculation)

The computations comprise an 12-bit multiplier and an accumulative adder. Note that 12 bits are needed to accumulate the multiplication of 240 inputs with their weights. The inputs to the activation function is defined by a Look-Up Table (LUT) storing the useful values. Apparently, we need 16 bits for the input of this function.

3.1 Register Transfer Level (RTL) description of MLP

The proposed parallel architecture describes a kind of layer parallelism with serial processing unit, in the sense that it requires one MAC unit per neuron for each layer ,and one sigmoid function for each layer. With this strategy, all the neurons of a layer work in parallel and therefore get their outputs simultaneously. This is not a fully parallel strategy because the outputs for different layers are obtained in a serial manner. For our particular MLP, where 20 neurons exist at a hidden layer and 10 at the output layer, 20 MAC units are required for hidden layer and 10 MAC units are required for output layer, **Figure.(5)** shows the basic structure of the MLP.

Given that all MAC units work in parallel for each input signal, they need access to the associated weights simultaneously, and hence, the weight RAM should be private for each one (as shown in **Figure.(5).** For this reason, the 8-bit counter is used to address 20 RAM modules containing 240 weights (as indicated in Figure). Since the data transmission between layers is serial, every MAC unit will also need some local storage for output data. Twenty parallel registers is used instead of the RAM module for the serial version, since this option reduces the writing time. The data in these parallel registers are introduced to the single activation unit by a 20:1 multiplexer whose select signals are by the 5-bit counter outputs. The output of the activation unit is used as a synaptic signal for the 10 neurons at the final layer.

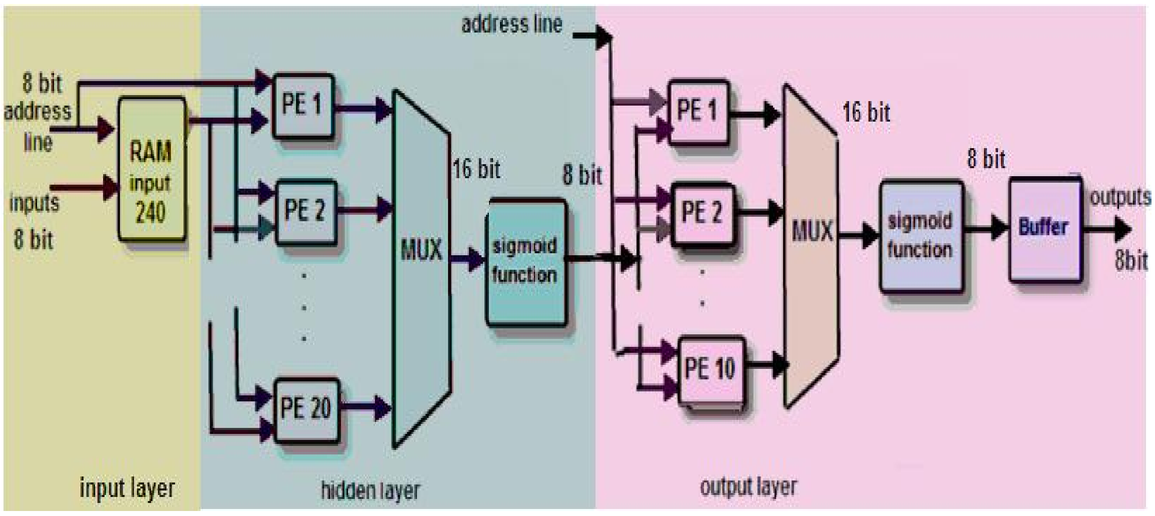

Fig. (5) Computing model structure of MLP for speech recognition

3.2 Register Transfer Level (RTL) description of Sigmoid Function

The hardware implementation of sigmoid function is a very important part of MLP hardware implementation. Because sigmoid function should complete a process of nonlinear mapping and includes several types of operation such as exponent, addition and division, much time and large hardware resource should be consumed if it is realized directly. So in this paper, sigmoid function is implemented by LUT in hardware. From the point of view of hardware implementation, the methods of sigmoid LUT generating are associated tightly with amount of memory unit that would be used and the complex degree of timing control is decided by the way of look up address generating. The implementation method introduced in [8-11], it is found that if input of sigmoid function exceeds 127, the output of sigmoid should be close to 255; if input of sigmoid function is less than 128, the output of sigmoid should be close to 0. If input belongs to range from -128 to 127, the output of sigmoid is calculated by equation(1) [3].

$$a = \frac{255}{1 + e^{-n/24}} \quad \text{..............................} \quad (1)$$

Where n is a positive integer

The result is saved to LUT. This method of sigmoid hardware implementation is suitable for both hidden layer and output layer. It also has been proved that MLP which uses this method to implement sigmoid, has same performance as neural networks that implement sigmoid function directly.

4. Results and Discussion

In this paper, an on-chip working process of MLP is implemented on FPGA , and off-chip training process is executed by software. The MLP based speech recognition system implements the parallel hidden and output layers with 20 and 10 serial neurons respectively. Although the study of the hardware resources consumption has a significant impact on the final product cost, any of the evaluated approaches would fit the application performance and specifications. For the speech recognition application, we obtained a correct classification rate of 90% with a computation time around 5.5 and 6.5 times faster than the software version running on 2.4 GHz Intel processor PC, which fulfilled the time restrictions imposed by the application. Therefore, the presented implementation can be seen as a low cost design: the whole system would fit into a low-cost FPGA device and could be embedded in a portable speech recognition platform for voice controlled systems. A pipeline processing scheme, using two neural layer, would lead to a faster approach. The processing bottleneck is imposed by the maximum neural fan in, because it needs 220 multiplications. With a pipeline structure, we can overlap the computation time of the hidden layer with the computation time of the output layer, which would speed up the data path to a maximum of 30% because the pipeline technique is regarded as a type of computation parallelism [see 3].

The results of implementation of MLP can be characterized by the following parameters[4]: number of slices, maximum clock rate, and data throughput (DTh) as the number of evaluated input vectors per second. In order to present the implementation characteristics of the designs with VHDL, we estimate the number of system gates (SGs) and the data throughput (DTh). Furthermore, to better illustrate the trade-of between these two characteristics, which can be adopted as a convenient design objective during the architecture definition process, we have evaluated the performance cost as Pc = SGs/DTh. In this way, we can evaluate how much hardware cost is required by a given performance. Table 1 shows the implementation results obtained after synthesizing MLP defined at a RTL abstraction level. The data throughput (DTh) is high due to the reduced number of clock cycles needed for each input vector evaluation. In this way, we are taking advantage of the inherent parallelism of the

ANN computation scheme. The reduction in the maximum clock frequency is due to the maximum combinational paths of the design.

Table 1 Hardware characteristics of the implemented MLP.

Hardware resources	
#Slices	4720
#System gates SGs	403005
Performance	
Max. Freq. (MHz)	17.6
# Cycles	270
DTh (data/s)	68300
Performance cost (Pc)	5.9

Although the final implementation characteristics depend on the designer's skills, and it is important that the speed of process and hardware resource is balanced well. The fully mapped prototype design onto the targeted Spartan3E XC3S500E FPGA [10,11,12] can be seen in **Figure. (6).**

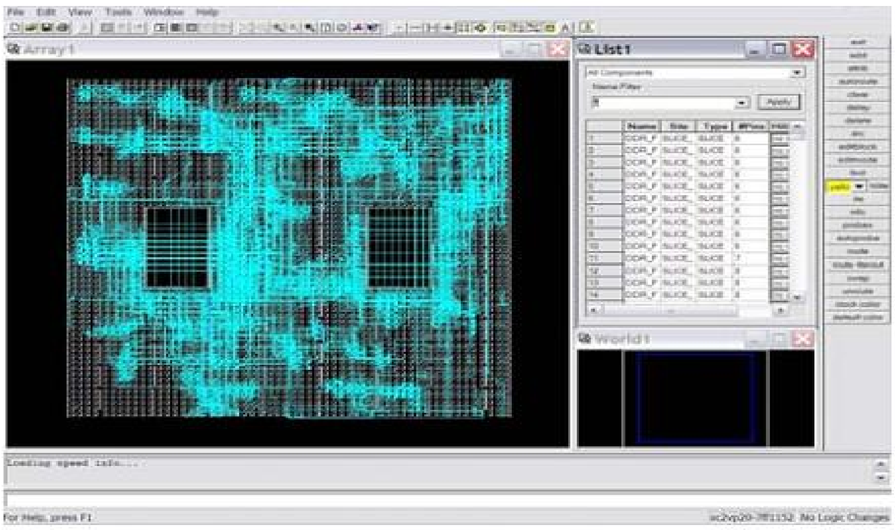
File Edit View Tools Window Help
Array1
List1
World1
Apply
Loading speed info...
For Help, press F1
xc2vp20-ff1152 No Logic Changes

Fig. (6) Placed and routed MLP with USB interface with Spartan3E FPGA kit.

The Spartan-3E Kit [10-11] includes embedded USB-based programming logic and an USB endpoint with a Type B connector. Via a USB cable connection with the host PC directly programs the FPGA **Figure (7). Figure.(8)** and **Figure.(9)** show the time diagrams of RAM unit waveforms and calculations of PEs waveforms for one input/output vector respectively.

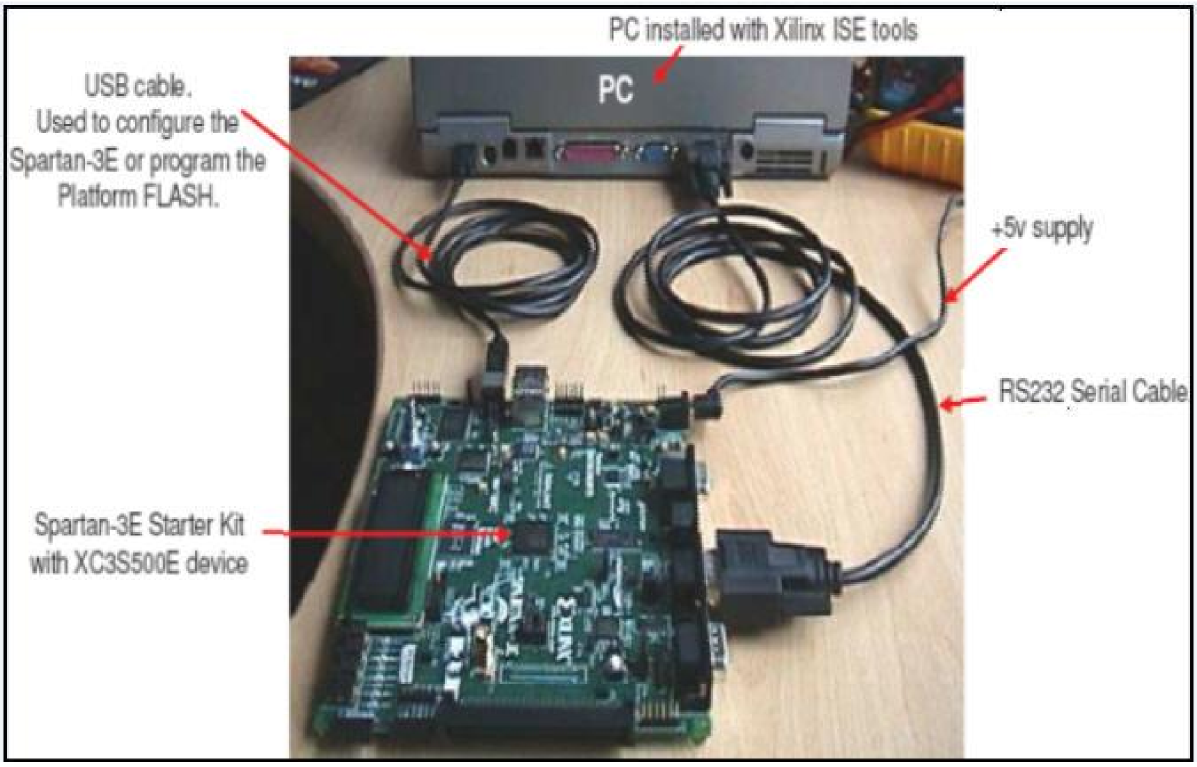
PC installed with Xilinx ISE tools
PC
USB cable. Used to configure the Spartan-3E or program the Platform FLASH.
+5v supply
RS232 Serial Cable
Spartan-3E Starter Kit with XC3S500E device

Fig. (7) The equipments needed for hardware implementation.

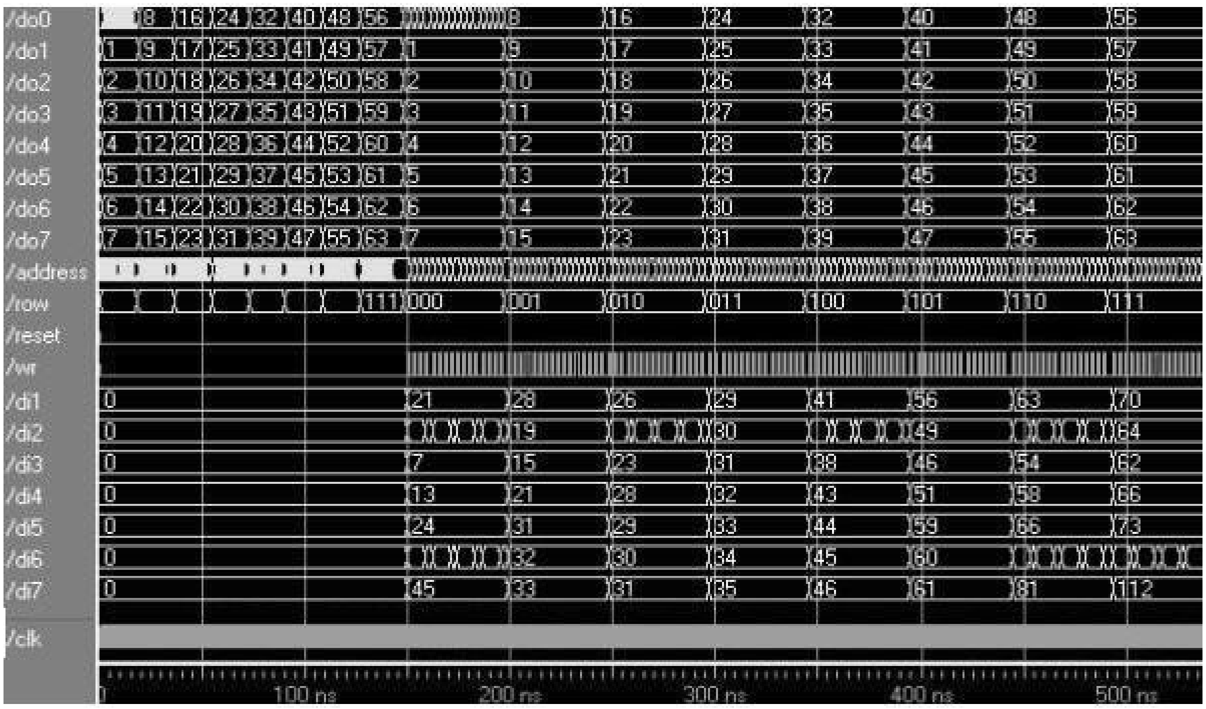

Fig. (8) The time diagram of RAM unit wavefourm.

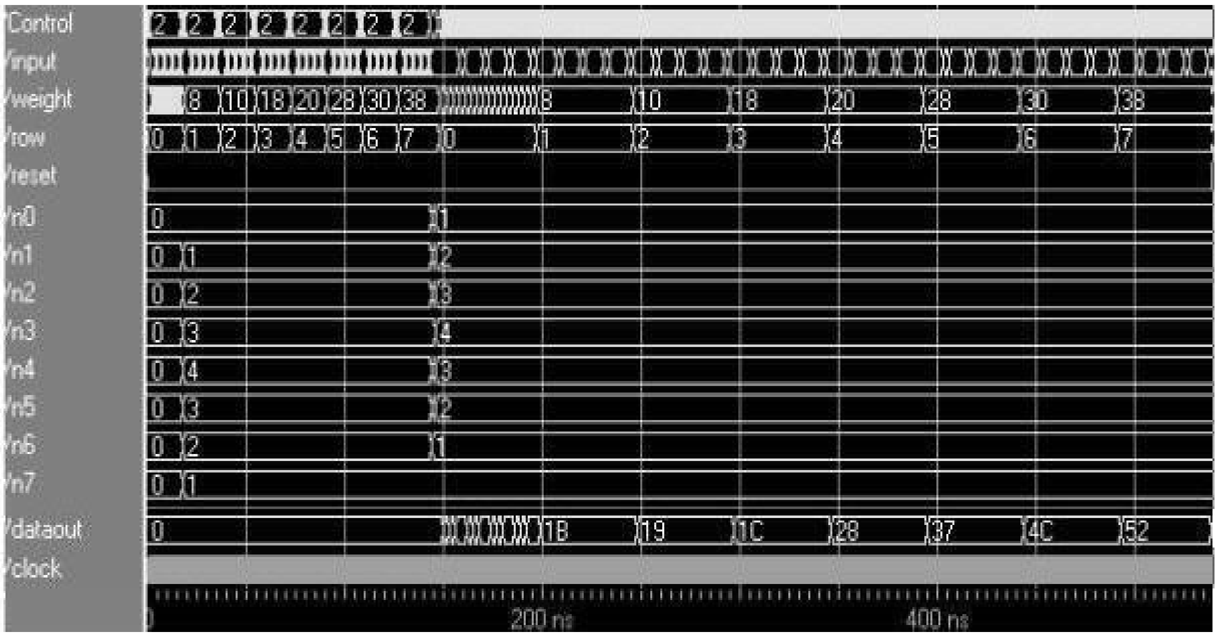

Fig.(9) The time diagram of PEs waveform for one input/output vector.

5. Conclusions

A mixed hardware architectures were presented to implement the MLP on Spartan3E XC3S500E FPGA chip. The main contribution of this work is the development of the mixed parallel neurons layers and serial MAC PE design. The design attempt to establishing a trade-off between chip-area and performance. The proposed computing architecture can accelerate the application of MLP in the embedded systems. We obtain a speaker-independent correct classification rate of 90% with a computation time 6 times speed up over the software implementation. Therefore, the implementation can be seen as a low-cost design where the whole system, would fit into single FPGA chip. The system could be embedded in a portable speech recognition platform for voice-controlled systems.

References

1. **Savich A., Moussa M., Areibi S., "The Impact of Arithmetic Representation on Implementing MLP–BP on FPGAs : A study", IEEE Transaction on Neural Network, Vol.18, No.1 , JANNARY 2007, PP. 240 – 255.**
2. **V. Pandya, S. Areibi , M. Moussa, "A Handel-C Implementation of the Backpropagation Algorithm On Field Programmable Gate Arrays", IEEE Proceedings of the International Conference on Reconfigurable Computing and FPGAs, 2005.**

3. **Xiaobin M., Lianwen J., Dongsheng S., Junrun E., "A mixed parallel neural networks computing unit implemented in FPGA", IEEE Int. Conf. Neural Networks & Signal Processing, Nanjing, China, December 14-17, 2003.**
4. **J.L. Rosselló, V. Canals and A. Morro "Hardware implementation of Neural Networks", In Proc. International Join Conference on Neural Networks (IJCNN 2010), Barcelona, 2010.**
5. **J.L. Rosselló, V Canals, A.Morro," Probabilistic-based Neural Network Implementation", IEEE world congress on computational intelligence,2012.**
6. **Sarvda Chauhan, Vikas Goel and Shalini Dhingra ,"Pattern Recognition System using MLP Neural Networks", IOSR Journal of Engineering, Vol. 2(5) , pp: 990-993, May. 2012.**
7. **R. Omondi, C. Rajapakse," FPGA Implementation of Neural Networks", Springer U.S., 2006,ISBN 10-0-387-28485-0.**
8. **Volnei A. Pedroni, "Circuit Design with VHDL", MIT Press Cambridge, London, England, ISBN 0-262-16224-5, 2004.**
9. **R. Lippmann, E. Martin, D. Paul, "Multi-style training for robust isolated word speech recognition", IEEE Transaction on speech and audio processing, vol. 3, no.4, July 1995.**
10. **Xilinx, "Spartan™-3E Platform FPGAs Complete Data Sheet", Data Sheet DS031-4 (v2.0), Xilinx, Inc., 2006.http://www.xilinx.com/**
11. **National Instruments Co ,"FPGA Fundamentals", May 03, 2012.**
12. **S. Kilts, "Advanced FPGA Design: Architecture, Implementation, and Optimization", John Wiley & Sons, Inc., Hoboken, New Jersey, 2007, ISBN 978-0-470-05437-6.**