Available online at www.sciencedirect.com

ScienceDirect

AASRI Procedia 6 (2014) 82 – 88

AASRI Procedia

www.elsevier.com/locate/procedia

2013 2nd AASRI Conference on Computational Intelligence and Bioinformatics

Design and Implementation of Multilayer Perceptron with On-Chip Learning in Virtex-E

Subadra Murugan[a]*, Packia Lakshmi K[b$], Jeyanthi Sundar[c], MathiVathani K[d]

[a]*Professor, Department of ECE, Einstein College of Engineering,Tirunelveli, Tamil Nadu, India-627 012.*
[b]*Assistant Professor, Department of ECE, Einstein College of Engineering,Tirunelveli,Tamil Nadu, India-627 012.*
[c,d]*PG Scholar,Applied Electronics, Department of ECE, Einstein College of Engineering,Tirunelveli, Tamil Nadu, India-627 012,.*

Abstract

Due to advancements in technology, many integrated circuits are fabricated to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain. Many of them use off-chip learning method either by analog hardware or massively by parallel computers. This proposed work is about a trainable neural chip using Field Programmable Gate Array (FPGA) as this helps in learning capability by exploiting the inherent parallelism of neural network. By this fast prototyping is possible for real-time applications, such as speech recognition, speech synthesis, image processing, pattern recognition and classification. In this work on-chip learning method is designed for standard benchmark XOR problem using back propagation based multilayer perceptron and is implemented in VIRTEX-E FPGA using VHDL. The design works at 5.332 MHz and the total gate count is 4, 73,237.

Keywords: FPGA; On-chip learning; Back propagation; Multilayer perceptron.

* Subadra Murugan.. Tel.: +0-979-096-8632; $ Packia Lakshmi K
E-mail address: *subadra_m@yahoo.com, $krishbagya@gmail.com

doi:10.1016/j.aasri.2014.05.012

1. Introduction

One of the emerging applications of Very Large Scale of Integration (VLSI) is standalone neural network chip. This stand alone neural network chip could surpass the capabilities of conventional computer-based pattern recognition systems and they are used in pattern classification, data processing, electrical load forecasting, power control systems, quantitative weather forecasting, games development, optimization problems etc. [1]. Artificial Neural Networks (ANNs) are powerful tool for modeling especially when underlying data relationship is unknown. It offers a completely different approach to solve the real-time problems and they are known as sixth generation of computing techniques [2].

Most of the existing neural network applications in commercial use are normally developed by software and sequentially simulated on a general-purpose processor [3]. This is the easiest but least favoured method, as the time taken for training is long. It is suitable for the networks which need not to be adapted to new data and used only for investing the capability of modeling the network [4]. However, there is some specific real-time application such as streaming video compression, bioinformatics which demands high volume adaptive real-time processing and learning of large non-linear dataset within a stipulated time [5]. Therefore, it necessitates the design of neural network hardware with truly parallel processing capabilities and reconfigurability for future extendable applications [6]. This can be achieved by on-chip learning method. It is a desirable method since it develops a way to make a stand-alone neural network chips. In on-chip learning method, hardware keeps itself ready to fix the architecture depending on the weight, to obtain the required performance by taking the full advantage of their inherent parallelism and runs faster in the order of magnitude than software simulation [7].

Chandrashekhar et al. (2013) compared the software and hardware implementation methods. Particularly they compared different hardware implementation methods like VLSI (analog/digital), Application Specific Integrated Circuit (ASIC) and FPGA. They have chosen FPGA implementation for their work because of high degree of programmability [8]. Suhap Sahin et al. (2006) compared the hardware implementation method using VLSI neural chips and FPGA. They have explained the usage of FPGA over VLSI chips for real-time applications. They described the FPGA features and concluded that FPGAs has higher speed and smaller size for real-time application than the VLSI design especially for classification [9]. Janardan Misra et al. (2010) also analyzed the different hardware implementation like analog neural chip, digital neural chip, RAM-based implementation method and FPGA implementation method. Finally they have concluded that FPGAs are low cost, readily available, and reconfigurable offering software like flexibility. They have further stated that the partial and online reconfiguration capabilities in the latest generation of FPGAs offer additional advantages [5].

This paper aims at the design of on-chip learning Multilayer Perceptron (MLP) based neural network with Back Propagation (BP) algorithm for learning to solve XOR problem. Section 1.1 reviews the architecture and Section 1.2 describes the learning algorithm of neural network. Section 2 deals with the implementation of on-chip learning followed by discussion of results.

1.1. Network architecture

Multilayer perceptron is one of the widely used universal approximator and is suitable to solve the non-linear separable problem [10]. Architectural parameters such as the number of inputs per neuron, weights, activation function, synaptic interconnection and number of layers are to be specified in ANN structure as they play significant role while learning [11]. Logical XOR function has two inputs and one output based upon which MLP is structured as in Fig 1.The ANN topology requires solving a non-linearly separable

problem consisting of at least one hidden layer [7]. Hence, in this work 2-2-1 MLP structure is designed with sigmoid activation function to solve XOR problem.

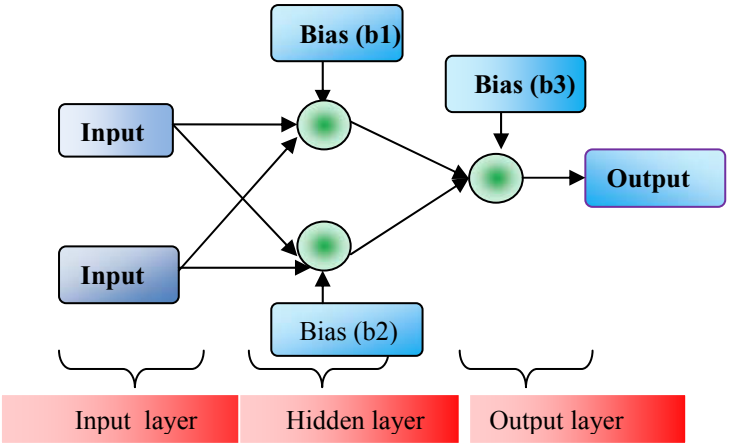

Fig. 1. Architecture of MLP for XOR problem

1.2. Learning process

The main objective of learning is to achieve good generalization that is able to predict the output values that are associated with a given input value [12]. The back propagation method is a supervised learning algorithm that is widely used for training the multilayer perceptron. It adjusts the weight values that are calculated from input-output mappings and minimize the error between the correct output value and target value. It iteratively computes the values of weight using gradient descent algorithm [10]. For the MLP with input vector x_i, output vector y_j the weight is calculated by gradient descent rule and is given by (1)

$$\Delta wi = \alpha\ yj(1-yj)\ (tj-yj)\ \ xi \qquad (1)$$

where, α is learning rate, y_j $(1-y_j)$ is the derivative of activation function and (t_j-y_j) is error with t_i as target. BP algorithm is based on repeated application of two passes namely- Forward pass and Feedback pass.

1.2.1. Forward pass

In forward pass, the input data travels from the input layer to output layer to obtain the output value at each processing element and the corresponding error value is calculated at the output layer. The output error computation is calculated as the difference between the actual output (y1, y2, … yn) and the desired output (t) and is calculated by equation (2) and (3)

$$\delta k = Yk\ (1-yk)\ (tk-yk)\ \ \textit{for each output neuron(k)} \qquad (2)$$

$$- \delta h = Yh\,(1-yh)\,\Sigma kwh,k\,\delta k \text{ for each hidden unit}(h) \quad (3)$$

1.2.2. Feedback pass

In this step the error value is propagated backward through the network, layer by layer in order to update the weight for the expected output. For each network weight $w_{i,j,}$ weight updation can be calculated as (4)

$$wi,j(new) = wi,j(old) + \Delta wi \quad (4)$$

where $\Delta w_{i,j} = \eta\ \delta_j\ x_{i,j}$ which is called as magnitude of weight updates. If the output is correct (t=y) the weights are not changed ($\Delta w_{i,j} = 0$) otherwise the weights w_i are updated. This training process is repeated until the output error signal falls below a predetermined threshold value or Mean Squared Error (MSE).

2. Implementation

As the logical XOR problem is used to benchmark the learning ability of ANN, this work utilizes on-chip propagation learning algorithm for solving XOR using VIRTEX FPGA. Fig 2 shows the complete BP algorithm implementation module. Forward pass is controlled by forward phase controller and backward pass of the learning process is controlled by backward phase controller. Global controller is used to synchronize the all modules.

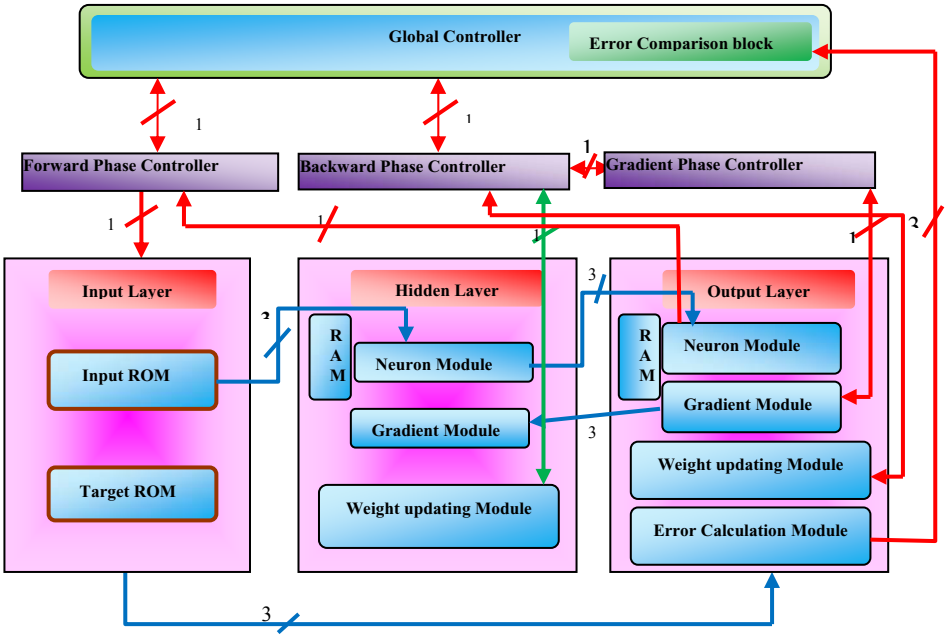

Fig. 2. BP algorithm implementation module

Each neuron in all layers is stimulated simultaneously and forwards the output to next layer. Finally the output is produced at the output layer by applying the activation function to calculate the value of weighted sum. This final output is given to the error calculation module to find MSE. Error comparison module of global controller is used for checking the condition of learning. If the error value is greater than the threshold value, global controller sends enabling signal to backward phase controller. The backward phase of BP algorithm consists of two main important phase

- Gradient calculation phase - calculation of new weight using gradient descent method
- Weight updating phase - new weight value is updated in all layer neurons

At first gradient value of output neuron is calculated and back propagated to previous hidden layer for its gradient calculation. After the gradient calculation, weight updating process takes place simultaneously for each layer and every weight of neuron. The new updated value is stored in RAM for future use. Thus completion of weight updating process indicates one complete training set and these two stages are repeated for all other input patterns until the network is sufficiently trained.

VHDL code is written for complete module of XOR problem using IEEE 754 standard single precision floating point arithmetic package and sigmoid activation function package. Hardware implementation of activation function is done using the approximation function using the following equation (5) [13].

$$f(net) = 1/22\left[\frac{net}{|net|} + 1\right] \quad (5)$$

3. Implementation Results

For on-chip training of XOR problem, the complete module is coded using VHDL and realized in VIRTEX -E using Xilin14.5 ISE. Once the design is completed, the top module is synthesized and verified for its timing and functionality. The Fig 3 & Fig 4 illustrates the training phase of the neural network in forward and backward pass respectively. From the figure it can be noted that the weight updating process is completed according to the error calculated in forward phase. The MSE ensures that the network is sufficiently trained. The resource utilization is given in the Table 1.

Table 1. Device utilized by the complete XOR module

Logic Utilization	Used	Available	Utilization
Total Number Slice Registers	2,851	64,896	4%
Number of occupied slices	32,015	32,448	98%
Number of slice containing only related logic	32,015	32,015	100%
Total Number 4 input LUTs	61,386	64,896	94%
Number used as logic	56,958	--	--
Number of GCLKs	4	4	100%
Number of GCLKOBs	1	4	25%
Total equivalent gate count for design	473,237	--	--

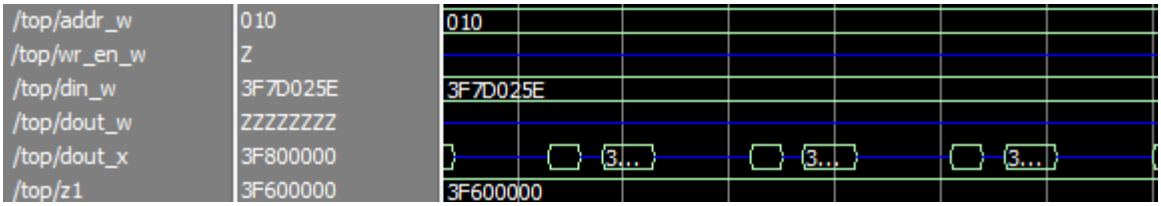

Fig. 3. forward phase

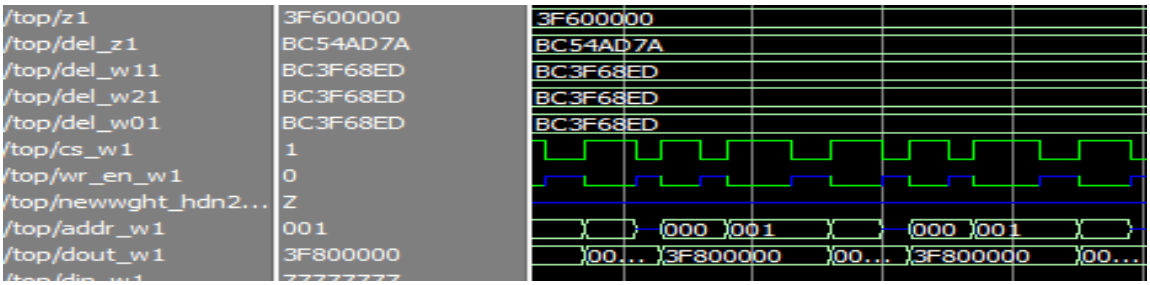

Fig. 4. backward phase

4. Conclusion

FPGA retains a high degree of flexibility for device reconfiguration and reduces the hardware development cycle. The proposed work is coded using VHDL and successfully simulated in MODELSIM 6.3f simulator and implemented in VIRTEX E FPGA using XILINX ISE 14.5 ISE. This work sustains the internal parallelism of artificial neural network and the design works at 0.6053315 µs. The design have utilised 87% of LUTs and 98% of slices. The result showcase the suitability of enhancing the MLP architecture with back propagation learning that can suit for real-time applications. Future work will be focused on designing the complex ANN to solve real-time problems.

Acknowledgements

This work is supported by the Department of Science and Technology (DST), Government of India under SERB Fast track Scheme for Young Scientists. The authors would like to acknowledge DST, India and Management and Principal of Einstein College of Engineering, Tamil Nadu, India for providing the facility and necessary support to design and implement this work.

References

[1] Ayman Youssef, Karim Mohammed, Amin Nassar. A Reconfigurable, Generic and Programmable Feed Forward Neural-network. IEEE 14th International Conference on Modelling and Simulation: 2012.
[2] Pinjare, Arun Kumar M. Implementation of Neural Network Back Propagation Training Algorithm on FPGA. International Journal of Computer Applications, 2012: (0975 – 8887).
[3] Packia Lakshmi.K, M. Subadra, PhD. Design and Realization of FPGA based Off-Chip Trained MLP for

Classical XOR Problem and Need of On-Chip Training. Special Issue of International Journal of Computer Applications (0975 – 8887) on International Conference on Electronics, Communication and Information Systems (ICECI 12): 2012.
[4] Jagath C, Rajapakse & Amos R. Omondi. FPGA implementation of Neural Networks". ISBN-10 0-387-28487-7, Spriger: 2006
[5] Janardan Misra, Indranil Saha b. Artificial neural networks in hardware: A survey of two decades of progress. Elsevier, Neurocomputing 2010: pp:239–255.
[6] Lorena P. Vargas1, Leiner Barba, Torres & L Mattos. Sign Language Recognition System using Neural Network for Digital Hardware Implementation. Journal of Physics: Conference Series: 2011.
[7] Packia Lakshmi.K , M. Subadra, PhD. A survey on FPGA based MLP realization for on-chip learning", International Journal Of Scientific & Engineering Research. 2012; Volume 4:ISSN 2229-5518.
[8] Chandrashekhar Kalbande, Anil Bavaskar. Implementation of FPGA based general purpose artificial neural network. ITSI Transactions on Electrical and Electronics Engineering.2013: 2320 – 8945.
[9] Suhap Sahin, Yasar Becerikli, Suleyman Yazic. Neural Network Implementation in Hardware Using FPGA. Springer-Verlag Berlin Heidelberg, ICONIP, 2006:1105 – 1112.
[10] Faycal benrekia, mokhtar & mounir bouheda. Gas sensor characterization and multilayer perceptron (MLP) hardware implementation for gas identification using a field programmable gate array (FPGA) programmable Gate Array(FPGA). Sensors journal 2013: pp: 2967-2985.
[11] Medhat Moussa and Shawki Areibi & Kristian Nichols . Arithmetic precision for implementing BP networks on FPGA", Springer, 2006 : pp.37-61.
[12] Kuno Kollmann, Karl-Ragmar Riemschneider , Hans Christoph Zeidler . On-Chip Backpropagation Training Using Parallel Stochastic Bit Streams. IEEE:1996.
[13] Thamer & khammas. Implementation of a sigmoid activation function for neural network using FPGA. 13th scientific conference of al-ma'moon uninersity college, Iraq; 2012.