Reconfigurable Artificial Neural Networks

Abhirup Basu
Student, ECE Dept. Brainware Group of Institutions-SDET, Kolkata, India

Pinaki Bisaws
Student, ECE Dept. Brainware Group of Institutions-SDET, Kolkata, India

Sarmi Ghosh
Student, ECE Dept. Brainware Group of Institutions-SDET, Kolkata, India

Debarshi Datta
Assistant Professor, ECE Dept. Brainware Group of Institutions-SDET, Kolkata, India

ABSTRACT
Artificial Neural Networks (ANNs) are highly parallel and interconnected under a single layer management system. Massive parallelism, distributed representation and computation and adaptability are the most typical characteristics of ANNs. Implemented ANN on Field Programmable Gate Array (FPGA) can be used for a variety of real life applications. FPGAs are more attractive devices for its reconfigurable architecture and lower power consumption than other processors. The lower non-recurring engineering (NRE) costs and short time to market for FPGAs are making it highly demand in hardware implementations. This paper proposes implementation of ANN architecture with feedforward network topology. To improve the speed of the system a LUT based activation function is implemented as a ROM which contains neuron synaptic weights and thus stores the inner product. The design has been synthesized and implemented on a Xilinx Spartan 6 target device using 14.7 ISE Design Suite and results are discussed. Design implementation of this proposed architecture is being enhance the overall performance of the system and as well as saving the area. The computation execution time of the proposed ANN architecture is 643.977 MHz which leads to fastest operation.

Keywords
ANNs, FPGA, LUT, Neural Network; VHDL

1. INTRODUCTION
Artificial neural networks (ANNs) are widely used in many domain such as signal processing, image and speech recognition, automation and control systems and medical diagnostic. Traditionally, ANNs are mostly designed in software for creating new algorithms. But software based simulation provides slower execution as compared with hardware based simulation. Hardware platform accelerate system performance for its inherent parallelism and hence, various neural networks topologies trend to implement hardware domain. FPGAs are one of the suitable hardware implementation for realization of ANNs as it conserves the parallel structural design of the neurons in a layer and provides flexibility in reconfiguration.

The high-speed, parallel data streams and I/O flexibility provided by today's FPGAs make it suitable choice instead of microprocessor. FPGAs offer several advantages with continuous improvement of VLSI technology. They have low power consumption, short time to market and high speed at lower prices. Implementing ANN on FPGA is the portable platform for several real-world applications. However, hardware implementation the ANN is not as simple as software implementation especially when considering a large number of neurons and the calculation of exponential function like activation function. There are several methods of activation function in ANN like piece-wise linear function, sigmoid function, hyperbolic tangent are used in ANNs architecture.

There are several researchers discussed to develop ANN structure implementing on FPGA. In [1] the authors described that hardware implementation of neural network is feedforward methodology with step activation function. The authors proposed algorithm for representing each neuron which exploits the special properties of Boolean function. In [2] the authors introduced a framework for implementing ANNs on FPGA. The proposed framework supports the multilayer perceptron type. In paper [10], the authors proposed the back propagation firing with help of neuron inputs and then it is multiplied with its corresponding input and weight associate with the input. Final value are summed together to get a multiple output. Each signal output is compared with the target value which is represented in error signal. In paper [11], the authors presented the ANN as an electrical circuit and the modeling are designed in Verilog. The circuit has been tested in Cadence tool. This system the circuit is represented as an axon and voltage is represented as dendrites of a cell. In paper [12], a very efficient and adaptable hamming ANN is proposed. This design is targeted to 0.35 μm CMOS process technology. This feed forward neural network has the ability to classify the neuron based on hamming distance of previously stored input neuron. VLSI implementation parallel neural network algorithm for test pattern generation is done in paper [13]. By using shift register mapping is done to VLSI array and programmable cells. Optimum hardware model can be reduced delay and phase shift in neural network which enriches the performance of the model [14]. Implementing ANN on FPGA by using multiplexing help to reduce the cost of resource utilization with moderate overhead of speed as discussed in paper [15].

In this paper, a modified ANN structure is described in VHDL and then it implemented on FPGA. The computation speed can be improved by using LUT based nonlinear function in the activation block. The related issues such as resource utilization, speed of computation are addressed through computational architectural solutions using FPGA.

2. ARTIFICIAL NEURAL NETWORK
ANNs are inspired biologically which can solve massively parallel computational problems. It is basically an information processing unit consists of number of interconnection processing elements (called neurons). The structure of artificial neuron consists of a number of input vectors, followed by multipliers (called weights) and then followed by an adder and at last an activation function is needed for providing a threshold value, as shown in fig 1. The ANN has n inputs, denoted as x_1, x_2,.....x_n. Each of this input

is associated with a weight, denoted as w_1, w_2,.....w_n respectively.

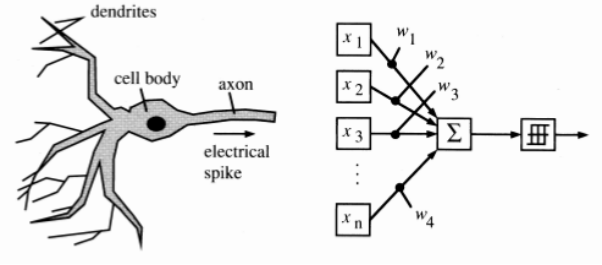

Fig 1: Feedforward Neural Network

In Biological Neural Network, a cell or neuron receives responses from other neurons via dendrites. When cell body contains enough action potential from dendrites, it fires and sends activation signal via axon to the dendrites of other neuron. In most general case, a neural network is a machine that is designed to model the way in which the brain performs a particular task or function of interest; the network is usually implemented using electronic components or simulated in software on a digital computer. In same manner ANN has performed its operation. It takes electrical signal from inputs and after addition all of these signals are passed through activation function. If the result of summation exceeds the threshold value than the output of activation function will be positive else, output of activation function will be negative. A positive value indicates excitatory connection while a negative value for a weight indicates an inhibitory connects.

An artificial neural network can be classified according to number of neurons in a layer, number of layer, orientation of neurons in a layer, type of feedback, etc. This work supports feedforward neural network. This network is simple and information transmits in one direction, forwards. Fig. 2 shows the architecture of feedforward neural network. In this layer the neurons get input from previous layer and forward output to the next layer.

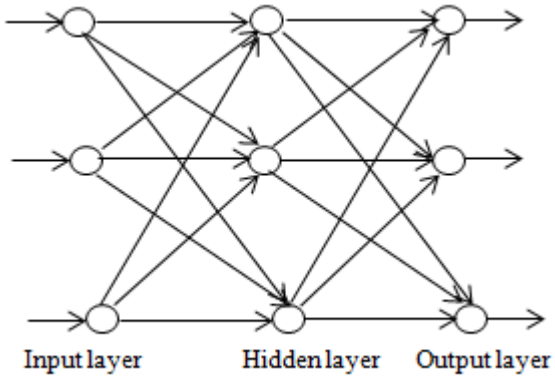

Fig 2: Feedforward Neural Network

3. HARDWARE ARVHITECTURE

ANNs are highly parallel and interconnected under a single layer management system. Parallelism, modularity and dynamic variation are the three most computational characteristics related with ANNs. ANN can be implemented using analog or digital systems. The digital implementation is more popular as it has the advantage of higher accuracy, better repeatability, lower noise sensitivity, better testability, higher flexibility, and compatibility with other types of preprocessors. Basically, there are three types to implement ANN structure in hardware platform. The implementation platform is DSP, ASIC and FPGA. The parallel structure of the ANN cannot be performed in DSP, since DSPs are inherent serial in nature. ANN structure can be implemented in ASIC platform but it cannot be modified to any further extent. Both of these problems can be solved in FPGA domain. The ANN implementation on FPGA aim to contribute in hardware integration with applied to different areas such as monitoring, diagnosis, maintenance and control of power systems. The FPGA implementation must ensure efficiency, timeless and a minimum possible space. The code is tested and implemented on a FPGA Spartan- 6 like by the ISE software from Xilinx.

Digital basic cell of ANN in feedforward with one output is shown in fig. 3. Implementation of feedforward ANNs for three neurons as inputs is described in HDL. The ANN architecture with three inputs and only one input weight which is sequentially shifted in each register.

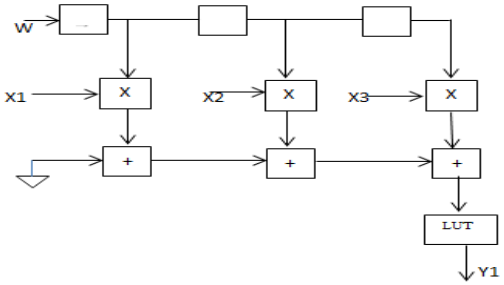

Fig 3: ANN implementation only one input for the weights

This configuration saving chip pins. The weights are then multiplied by the inputs and accumulated to get the intermediate outputs. These intermediate outputs are then applied to lookup table (LUT) which implement activation function and thus produces final outputs of the neural network. There are different types of activation function are used.

The most important parts of a neuron are activation function, especially for the nonlinear activation function. A high precision activation function is required for a high accuracy output of the ANN structure. The output of an activation function is connected to the next input of the ANN layer. If any error exists at output of the activation function that will be accumulated and amplified by the ANN and forwarded to next layer. Hence, a random result will be generated. To get an error free result, the sigmoid function is implemented as an activation function mostly used in neural networks applications. But it is not easily realized to implement efficiently. Sigmoid function has infinite exponential series, so that it is not suitable to implement on FPGA. Alternative approaches like look-up tables (LUT), piecewise linear approximation are usually used to implement on FPGA. This paper supports LUT based activation function for the approximation of the sigmoid function is shown in fig. 4. The transition region of the sigmoid function broken into several segments and stores it into a table. Basically LUT consists of ROM table which is addressed by the total synaptic input. Computation results are loaded into LUT which gives the output of neurons.

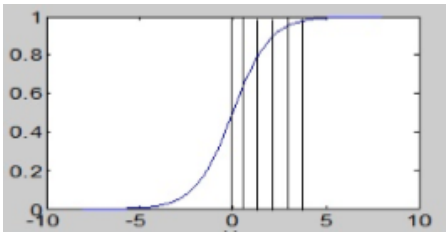

Fig 4: LUT based sigmoid function [17]

Fig. 4 shows the sigmoid function, approximately it is 1 when x=5 and 0 when x = -5. From the picture it shows that transition range is between -5 to 5. The sigmoid function is symmetry point at (0, 0.5) and hence only one half portion of this curve has to be computed.

4. EXPERIMENTAL RESULT

The simulation waveform of ANN architecture using Xilinx 14.7 simulator is as shown in figure 5. Here, proposed ANN architecture is three neurons with each neuron 4-bit inputs and shifted weights are multiplied the inputs and accumulated to produce the desired outputs. The architecture is codded using Very High Speed Integrated Circuits Hardware Description Language (VHDL). The VHDL-FPGA combination is very powerful embedded system design tool for its low cost and short time to market. The design is successfully validated using hardware simulation feature of Spartan-6 family of target device XC6SLX45, speed grade -3.

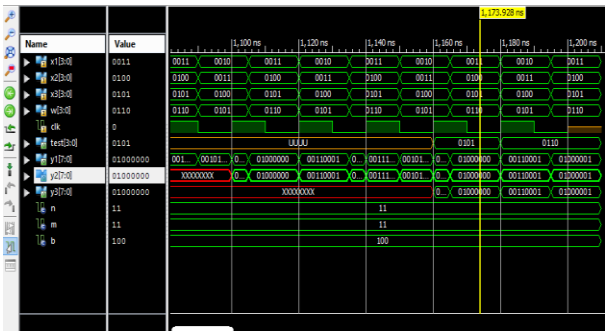

Fig 5: Simulation waveform of ANN architecture

Fig. 6 shows Register Transfer Language (RTL) of the proposed architecture. The resource utilization such as slices, logic elements and maximum operation frequency can be obtained in synthesis report. The computation speed of the proposed ANN architecture in feedforward topology is 643.977 MHz which leads to fastest operation. Table I represents the hardware resources of this architecture.

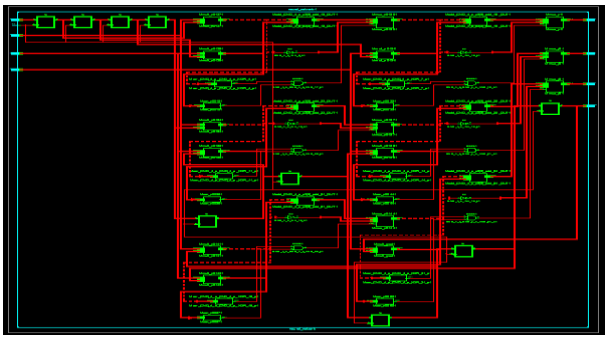

Fig 6: RTL Schematic of ANN architecture

Table 1- Device Utilization Summary of Proposed Neural Network Architecture

Synthesis Parameter	Proposed Architecture
Number Slice Registers	36
Number of Slice LUTs	255
Number used as Flip-Flops	36
Number used as logic	243
Number of IOBs	45

The device utilization summary demonstrates that the implemented architecture employs a few numbers of the slices are required for complete this project, which makes it suitable to develop a large scale implementation. This implementation must ensure efficiency, timeless and a minimum possible space on the FPGA.

4. CONCLUSION

In this paper hardware implementation of ANNs has been presented on FPGA as one can exploit concurrency and rapidly reconfigure to adapt the weights and topologies of an ANN. By using of FPGA take a high speed reconfigurable hardware device and as a result make it economic investment for designing ANNs. Each neuron is implemented a weighted ROM which is shifted sequentially and one Multiply Accumulator (MAC) unit. This value applies to LUT which implements the activation function and produces the final output. The proposed ANN architecture is efficiently programmed with reusable VHDL code. The proposed ANN architecture has been synthesized and implemented on Spartan 6 to target device XC6SLX45. The development ANNs on FPGA in this paper has revealed to a satisfactory solution in terms of computation time and hardware resources. The slices used in this proposed methodology are 36 which reflect that area required of this architecture is small. The computation execution time of the proposed ANN architecture is 643.977 MHz which leads to fastest operation. This project can be implemented in various real-time applications to obtain high speed with desirable accuracy. In future pipeline architecture is to improve the speed which is applicable in real time system.

5. ACKNOWLEDGMENTS

I would like to thank Brainware Group of Institutions for providing us XILINX ISE Design Suite 14.7 and FPGA board.

6. REFERENCES

[1] A. Dinu, M. N. Cirstea and S. E. Cirstea, “Direct Neural-Network Hardware-Implementation Algorithm,” IEEE Transaction on Industrial Electronics, Vol. 57, No.5, May 2010.

[2] H. K. Ali and E. Z. Mohammed, “Design Artificial Neural Network Using FPGA,” International Journal of Computer Science and Network Security, Vol. 10, No. 8, August 2010.

[3] S. Sahin, Y. Becerikli, and S. Yazici, “Neural Network Implementation in Hardware Using FPGAs,” LNCS 4234, pp. 1105 – 1112, 2006.

[4] V. A. Pedroni, "Circuit Design with VHDL," MIT Press, Cambridge, 2004.

[5] D. E. Khodja, A. Kheldoun and L. Refoufi, "Sigmoid Function Approximation for ANN Implementation in FPGA Devices," Recent Researches in Circuits, Systems, Electronics Control & Signal Processing ISBN: 978-960-474-262-2

[6] S. Moukhlis, A. Elrharras and A. Hamdoun, "FPGA Implementation of Artificial Neural Networks," International Journal of Computer Science Issues, Vol. 11, Issue 2, No 1, March 2014.

[7] M. Skrbek, "Fast Neural Network Implementation", Neural Network World, Vol. 9, No. 5, pp. 375–391, 1999.

[8] J. G. Eldredge, "FPGA Density Enhancement of a Neural Network Through Run-Time Reconfiguration",Master's thesis, Department of Electrical and Computer Engineering, Brigham Young University, May 1994.

[9] K.P. Sridhar, B. Vignesh, S. Saravanan, M. Lavanya and V. Vaithiyanathan, "Design and Implementation of Neural Network Based circuits for VLSI testing" World Applied Sciences Journal 29 (Data Mining and Soft Computing Techniques): 113-117, 2014.

[10] N. Chhedaiya and V. Moyal, "Implementation of Back Propagation Algorithm in Verilog," Interantional Journal of Computer Technology and Applications, pp: 340-343, 2012.

[11] E. Gieva, R. P. Rusev, R. I. Radonov, T. B. Takov and M. Hristov, "Verilog-A Behavioral Model of Hydrogen Bonding Network," Annual Journal of Electronics, 2011.

[12] S. Badel, A. Schmid and Y. Leblebici, "A VLSI Hamming Artificial Neural Network with k Winner-Take-All and k Loser-Take-All," Capability International Joint conference on Neural Networks, Portland, 2003.

[13] M. Fayyazi and Z.Navabi, "Accelerating Test Generation by VLSI Hardware Emulation," The 9th IEEE North Atlantic Test Workshop, 2000.

[14] Peter, B.L. Meijer, "Neural Networks for Device and Circuit Modelling," Lecture notes in computing science and engineering, pp: 251-258, 2001.

[15] Asha, R., P. Bowrna and F. Mhaboobkhan, "Implementation of Feed forward Neural network using layer multiplexing for effective resource utilization in FPGA," International journal of research in Engineering and Advanced Technology, 2013.

[16] R. Omondi, C. Rajapakse," FPGA Implementation of Neural Networks",ISBN 10-0-387-28485-0 Springer U.S., 2006.

[17] S. Ngah, R. A. Bakar, A. Embong and S. Razali, "Two-steps Implementation of Sigmoid Function for Artificial Neural Network in Field Programmable Gate Array," ARPN Journal of Engineering and Applied Sciences, Vol.11,No.-7,April,2016.