International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 5 | ISSN : 2456-3307

Design and Implementation of Logic Gates using Artificial Neural Networks on FPGA

Bharath Rao Madela, V. Yaswanth Siva Sai
Department of Electronics and Communication Engineering, SVNIT, Surat, Gujarat, India

ABSTRACT

In this paper, a hardware implementation of artificial neural networks and implementation of logic gates using artificial neural networks on Field Programmable Gate Arrays (FPGA) is presented. A digital system architecture for feed forward multilayer neural network is realized. The parallel structure of a neural network makes it potentially fast for the computation of certain tasks that makes a neural network well suited for implementation in VLSI technology. Then logic gates are implemented using Feed Forward Neural Network. FPGA has been used to reduce the unit neuron hardware by designing the activation function inside the neuron without the need of lookup tables. The whole design is realized using Verilog HDL and is implemented on FPGA.

Keywords: Artificial Neural Network, FPGA, Verilog, Activation Function, Feed Forward Propagation.

I. INTRODUCTION

Neural networks have been widely used in many fields, either for development or for application. They can be used to solve a wide variety of problems that are difficult to be resolved by other methods. They are mostly employed in Artificial Intelligence. [1]

Although neural networks are mostly implemented in software, hardware versions are gaining importance. Software versions have the advantage of being easy to implement, but with poor performance. Hardware versions are generally more difficult and time consuming to implement, but with better performance. [2]

In technological view, humans are trying to emulate the behaviour of Biological Neuron. ANNs are biologically inspired and require parallel computations in their nature. So Microprocessors and DSPs are not suitable for parallel designs. Designing parallel modules can be available by ASICs and VLSIs. In addition the design results in an ANN is only suited for one target application. FPGA's not only offer parallelism but also flexible designs, savings in cost and design cycle. [3]

The architecture is designed using Verilog Hardware description language.

II. STRUCTURE OF ANN

An artificial neural network is an interconnected group of nodes, which perform function collectively, and in parallel, akin to the vast network of neurons in a human being. It consists of a number of input vectors, followed by multipliers, which are often called as weights followed by a summer and an activation function. The input signals are summed and sent through a threshold function. If the result of the summation operation exceeds the threshold value, the neuron fires i.e. the output of the threshold function will be positive else, it will give a negative value. [4]

The artificial neuron model, which is shown in Figure 1, is widely used in artificial neural networks with some variations.

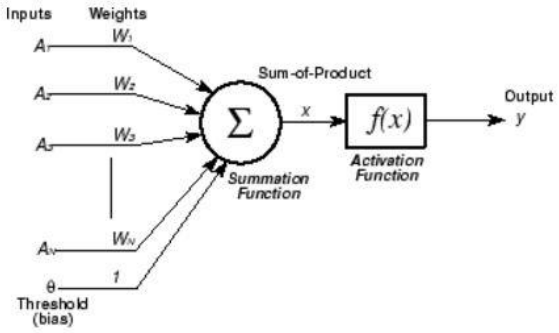

Figure 1. Structural diagram of a Neuron

The artificial neuron given in this figure has N inputs, denoted as A1, A2… An. Each line connecting these inputs to the neuron is assigned a weight, denoted as W1, W2….Wn respectively. The activation, x, determines whether the neuron is to be fired or not. It is given by the formula:

$$x = \sum_{j=1}^{N} A_j W_j \qquad (1)$$

A negative value for a weight indicates an inhibitory connection while a positive value indicates excitatory connection. The output, y of the neuron as a function of x is given by

$$y = f(x) \qquad (2)$$

The output function f(x) is known as a threshold function or activation function.

III. NEURON ACTIVATION FUNCTION

One of the important part of the neuron is its Activation function of a node defines the output of that node given in an input or set of inputs. The non-linearity of the Activation function makes it possible to approximate any function. The types of Activation function used in this work are symmetrical hard limit activation function, Saturating Linear Activation Function and Sigmoid Activation Function. [5]

A. Symmetrical Hard limit Activation Function

It is referred as 'Hardlims'. It is used to classify inputs into two distinct categories. Hard limiting means clipping, it is a limiting action in which there is a over permitted dynamic range, negligible variation in the expected characteristics of the output signal and steady state signal at the maximum permitted level. Hard limit activation function is given below.

$$x = \begin{cases} -1, & n < 0 \\ 1, & n \geq 0 \end{cases}$$

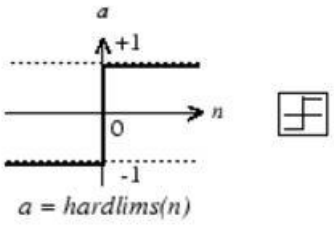

Figure 2. Symmetric Hard limit Activation Function

B. Saturated Linear Activation Function

It is also known as satlins. This transfer function calculates a layers output from its net input. The output is as shown below.

$$a = \begin{cases} -1, & n < 0 \\ 0, & -1 \leq n \leq 1 \\ 1, & n > 0 \end{cases}$$

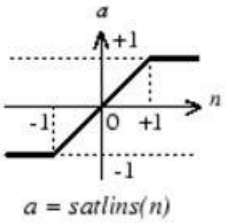

Figure 3. Saturated Linear Activation Function

C. Sigmoid Activation Function

In the hardware concepts of Neural Networks, it is not easy to implement on FPGA, because it consists of infinite exponential series. Formula to calculate sigmoid function is given as

$$y = \left(1 + e^{-(x-t)}\right)^{-1}$$

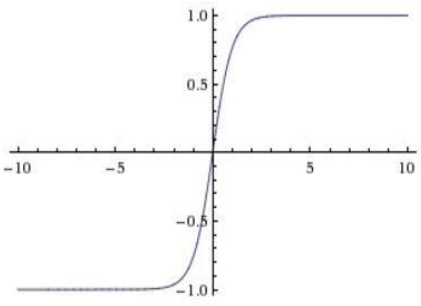

Figure 4. Sigmoid Activation Function

IV. FEED FORWARD SYSTEM OF ANN

The Structure of the neuron can be realized in many ways, mainly considering the degree of the parallel computation needed. The proposed Structural diagram for hardware implementation of neuron is shown in Figure 5. The structure contains two shift registers, one contains the weights while the other register holds the input. This approach is appropriate for general purpose neuron i.e. programmable neuron. It employs only one input to load all weights thus saving on chip pins. The weights are shifted sequentially until the register is loaded. The weights are then multiplied by the input and accumulated to produce the desired output [6]. Verilog HDL has been used for realizing the structure of neuron.

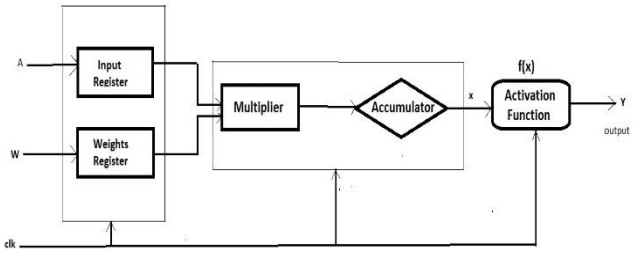

Figure 5. Structural Diagram of Neuron

The Feed Forward networks are generally arranged in distinct layers that contain only forward path that is shown in Figure6. In this type of network each layer receives inputs from the previous layer and outputting to the next layer which indicates there is no feedback. It means that signals from one layer are not transmitted back to the previous layer. [7]

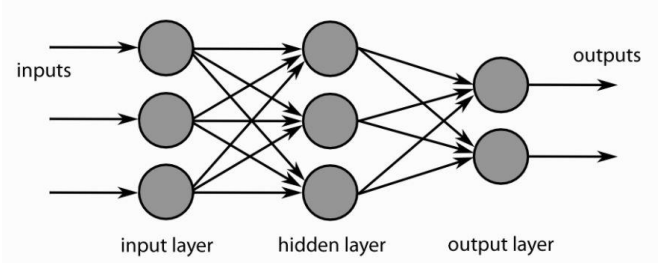

Figure 6. Feed Forward neural network

V. DESIGN OF LOGIC GATES USING ANN

Different logic gates have been implemented using the Feed Forward Neural Networks. Single bit inputs are taken and 4 bit weights are taken for the design. In neural networks, if output is greater than 0, it is considered as +1 and if the output is less than 0, it is considered as -1. The neuron model is like a universal gate which can be modified into other gates just by changing the appropriate weights.

The AND gate is implemented using the Hard limit activation function. In case of Hard Limit activation function a high output (1) results only if obtained output from the neural network is greater than zero. The low output (-1) results only if obtained output from the neural network is less than zero (0).

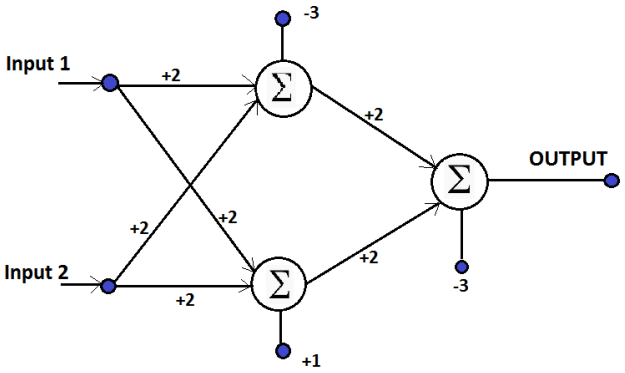

Figure 7. Neuron model for AND Gate with hardlims

The OR gate is a logic gate which performs logical disjunction. A High output (1) results if one or both the inputs to the gate are high. If neither of the inputs are high, a Low output (0) results. In neural network, if output is greater than 0 it is considered as +1, if output is less than 0 which is very small then it is considered as -1. This can be decided by the Saturating Linear Activation function.

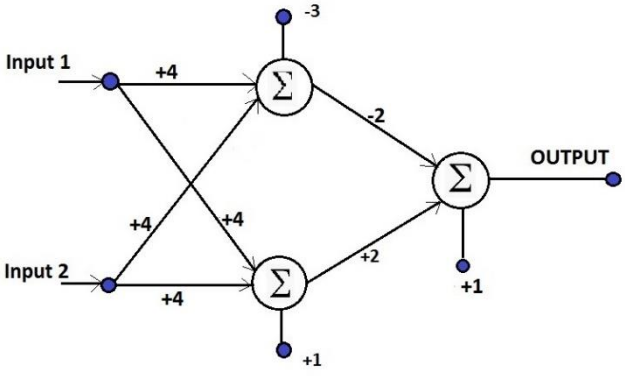

Figure 8. Neuron Model for OR Gate using Satlins

The XOR gate gives a High output (1) when the number of true inputs are odd i.e. When the 2 given inputs are complimentary to each other, a logic High (1) occurs. The XOR gate is realized by using sigmoid

function as the transfer function. The realization of XOR gate is shown in the Figure 9.

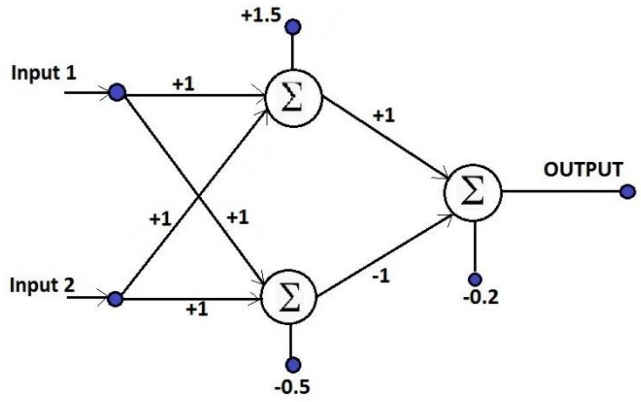

Figure 9. Neuron Model for XOR Gate using Sigmoid Activation Function

VI. RESULTS AND DISCUSSIONS

The logic gates using Artificial Neural Network models have been written in Verilog HDL and is implemented in Spartan-3e FPGA. The results from the Xilinx ISE tool are furnished below

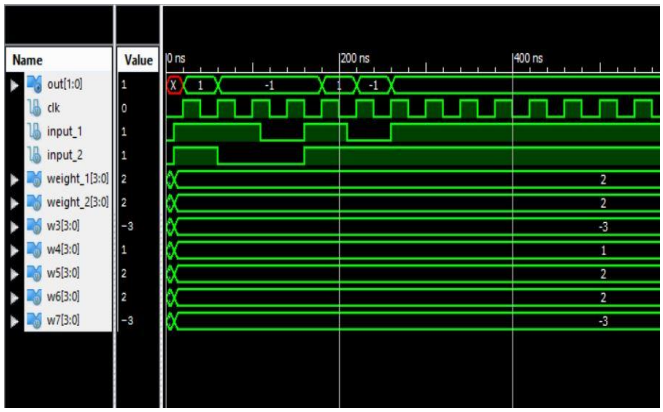

Figure 10. Simulation for AND Gate with Neuron model

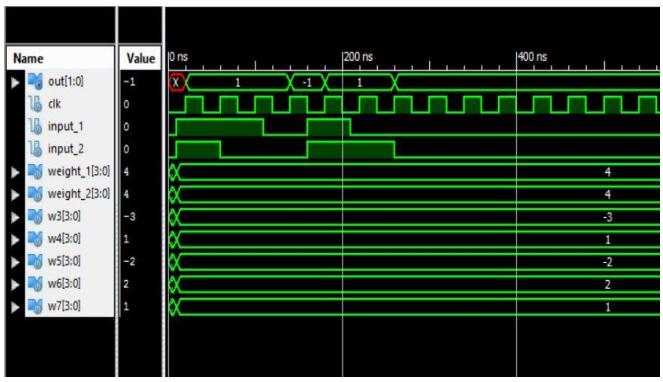

Figure 11. Simulation for OR Gate with Neuron model

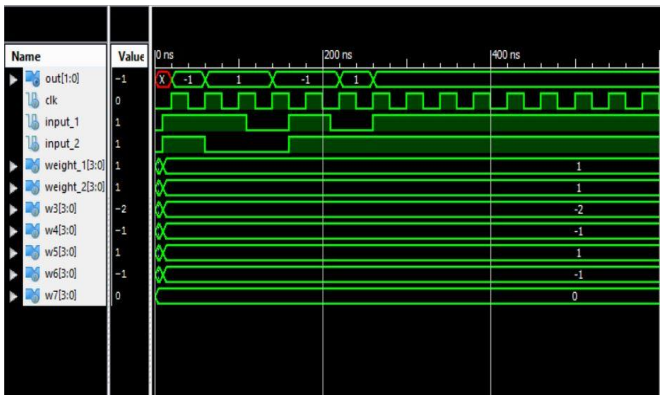

Figure 12. Simulation for XOR Gate with Neuron model

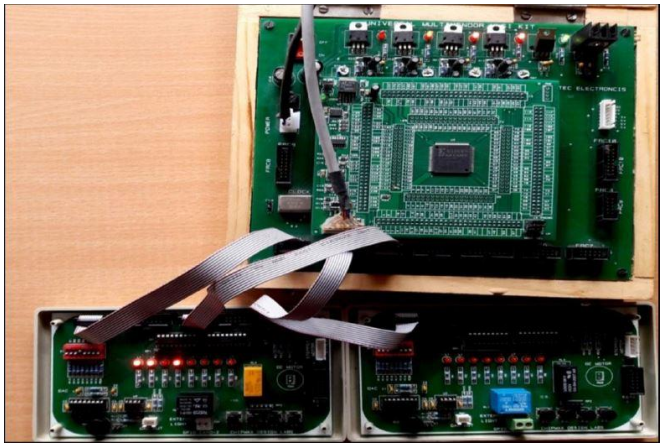

Figure 13. ANN Hardware implementation on Spartan-3 FPGA

VII. CONCLUSION

The logic gates using artificial neural networks with feed forward propagation have been successfully implemented. The hardware implementation offers a high parallelism and efficiency compared to software versions. As the field is of artificial intelligence is ever increasing, and thus requiring high degree of parallel computation which can be provided by hardware versions. Machine learning concept can be adhered to ANN by training the gates to learn by themselves by using Back Propagation methods.

VIII. REFERENCES

[1]. B. Widrow, D. E. Rumelhart, and M.A. Lehr, "Neural Networks: applications in industry, business and science," Communications of the ACM, vol. 37, no. 3, pp. 93-105, 1994.

[2]. Rolf F. Molz, Paulo M. Engel, Fernando G. Moraes, "Codesign to Fully Parallel Neural Network for a Classification Problem". University Montpellier II, France, 2000.

[3]. A. Muthuramalingam, S. Himavathi, and E. Srinivasan, "Neural network implementation using fpga: Issues and Application," The International Journal of Information Technology, vol. 4, no.2, pp.86-92, 2008.
[4]. Hardik H. Makwana, Dharmesh J. Shah, Priyesh P. Gandhi."FPGA Implementation of Artificial Neural Network" International Journal of Emerging Technology and Advanced Engineering Volume 3, Issue 1, January 2013.
[5]. Esraa Zeki Mhammed and Haitham Kareem Ali,"Hardware implementation of Artificial Neural Network Using Field Programmable Gate Array" International Journal of Computer Theory and Engineering, Vol. 5, October 2013.
[6]. Emmanuel Adetiba, F.A. Ibikunle, S.A. Daramola and A.T. Oliajide, "Implementation of Efficient Perceptro ANN Neurons on Field Programmable Gate Array Chip" International Journal of Engineering & Technology IJET-IJENS Vol 14 No:01.
[7]. A. Savran and S.Unsal, "Hardware Implementation of a feed forward neural network using FPGA's" Ege, Department of Electrical and Electronics Engineering, 2003.