# Implementation of Multilayer Feed Forward Neural Network using VHDL

Amitkumar B. Khonde
B. Jain Institute of Technology Management and Research
Nagpur

Yogesh Sharma
B. Jain Institute of Technology Management and Research
Nagpur

Sanjay Badjate, PhD
B. Jain Institute of Technology Management and Research
Nagpur

## ABSTRACT

In this paper a hardware implementation of a neural network NN using Field Programmable Gate Arrays (FPGA) is presented. A digital system architecture is designed to realize a feed forward multilayer neural network. The designed architecture is described using Very High Speed Integrated Circuits Hardware Description Language (VHDL) and implemented in an FPGA chip. The design is verified on an FPGA demo board Xilinx Spartan.

## Keywords

FPGA, VHDL,NN.

## 1. INTRODUCTION

Artificial Neural Networks have been widely used in many fields. A great variety of problems can be solved with ANNs in the areas of pattern recognition, signal processing, control systems etc. Most of the work done in this field until now consists of software simulations, investigating capabilities of ANN models or new algorithms. But hardware implementations are also essential for applicability and for inherent parallelism. There are analog and digital and also mixed system architectures proposed for the implementation of ANNs. The analog ones are more precise but difficult to implement and have problems with weight storage. Digital designs have the advantage of low noise sensitivity, and weight storage is not a problem. With the advance in programmable logic device technologies, FPGAs has gained much interest in digital system design. They are user configurable and there are powerful tools for design entry, syntheses and programming.

FPGAs not only offer parallelism but also flexible designs, savings in cost and design cycle. Considering the relevance of choosing FPGA in order to implement ANNs, the purpose of our work is to describe in VHDL an ANN dedicated to be implemented on a other FPGA kits like Xilinx Spartan kit. Selecting a simple Feed Forward Neural Network dedicated composed of three neuron in input one hidden layer with five neurons and one neuron in the output layer.

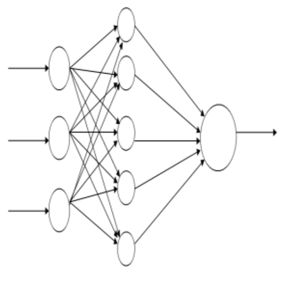

**Figure 1: Multilayer ANN**

## 2. ARTIFICIAL

An artificial neuron forms the basic unit of artificial neural networks. The basic elements of an artificial neurons are a set of input nodes, indexed by, say, 1, 2, 3, that receives the corresponding input signal or pattern vector, say x = (x1, x2, .x3 ); a set of synaptic connections whose strengths are represented by a set of weights, here denoted by w = (w1, w2, w3); and an activation function Φ that relates the total synaptic input to the output (activation) of the neuron. The main components of an artificial neuron is illustrated in Figure 2 The total synaptic input, ***u,*** to the neuron is given by the inner product of the input and weight vectors**:** ***u = €wi xi.*** where we assume that the threshold of the activation is incorporated in the weight vector. The output activation, *y*, is given by *y* = Φ(*u*). where Φ denotes the activation function of the neuron.

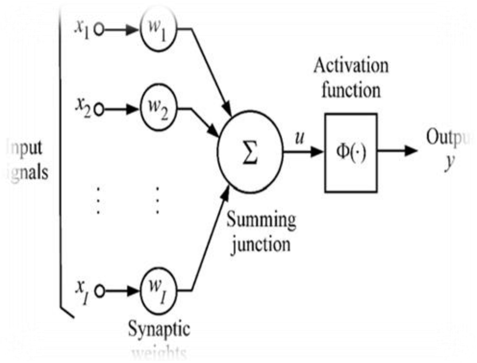

**Figure 2: The basic components of an artificial Neuron**

Consequently, the computation of the inner-products is one of the most important arithmetic operations to be carried out for a hardware implementation of a neural network. This means not just the individual multiplications and additions, but also the alternation of successive multiplications and additions — in other words, a sequence of multiply-add (also commonly known as multiply-accumulate or MAC) operations. We shall see that current FPGA devices are particularly well-suited to such computations.

## 3. NEURON ARCHITECTURE

The processing element of an ANN is the Neuron. A Neuron can be viewed as processing data in three steps; the weighting of its input values, the summation of them all and their filtering by sigmoid function. The summation can be calculated by a serial accumulation. For the weighted inputs to be calculated in parallel using conventional design techniques, a large number of multiplier units would be required. To avoid this, Multiplier/Accumulator architecture has been selected. It takes the input serially, multiplies them with the corresponding weight and accumulates their sum in a register.

The processes synchronized to clock signal. The number of clock cycles for a neuron to finish its work, equals to the number of connections from the previous layer. The accumulator has a load signal, so that the bias values are loaded to all neurons at start-up. Figure (3) shows the proposed neuron design.

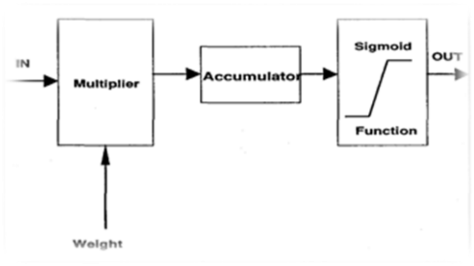

**Figure 3: Neuron Architecture**

## 3.1 Memories

Memories enable to stock data. In this particular PFGA implementation, only RAMs and ROMs were used. ROMs (Read Only Memory) stock data which can only be read, so we chose to implement two ROMs, one to stock the weight and one to stock the network parameters.

In a RAM (Random Access Memory) data can be updated and read, so we used one to stock the inputs of the networks that will be written directly through a test file and one for the outputs of each hidden layers that will be modified by the neurons themselves and read by the ones of the next layer.

## 3.2 Calculation unit

One of the most important parts of a neuron is its activation function. The nonlinearity of the activation function makes it possible to approximate any function.. This holds true for the nonlinear activation function used at the output of neurons. A common activation function is the sigmoid function

$$\mathbf{Y = 1/1+e^x}$$

Efficient implementation of the sigmoid function on an FPGA is a difficult challenge faced by designers. It is not suitable for direct implementation because it consists of an infinite exponential series. In most cases computationally simplified alternatives of sigmoid function are used.

## 4. SYNTHESIS AND IMPLEMENTATION

The Synthesis and implementation of this activation function and feed forward neural network with three neuron at input layer and five neurons at middle layer one at output layer is to be done in VHDL language on Xilinx 9.2 version. Following Table **1** give performance and resource use summary for all implemented 9 neurons. The operation speed in all cases gives good results and shows the advantages of using FPGAs in neural realization.

**Table 1**

| Device utilization summary (estimate value) | | | |
|---|---|---|---|
| **Logic utilization** | **used** | **Available** | **utilization** |
| Number of Slices | 1231 | 768 | 160% |
| Number of slice Flip flop | 792 | 1536 | 51% |
| Number of 4 input LUTs | 2116 | 1536 | 137% |
| Number of bonded IOBs | 82 | 124 | 66% |
| Number of GCLKs | 1 | 8 | 12% |

## 5. CONCLUSION

It has been proved that using FPGAs to implement ANN is a good option since VHDL offers many design possibilities and advantages when, at the same time, FPGA bring their flexibility and cost-efficiency. This work could be improved VHDL design on Xilinx kit board. So comparison in terms of execution speed and capacity with other FPGA kits like Raspberry Pi or Xilinx Spartan or Virtex series is performed in this paper.

## 6. REFERENCES

[1]. Philippe Dondon,v Julien Carvalho, Rémi Gardere, Paul Lahalle, Georgi Tsenov and Vale Mladeno "Implementation of a Feed-forward Artificial Neural Network in VHDL On FPGA '' Neural Network Application in Electrical Engineering (NEURAl) IEEE CONFERENCE 27 NOV 2014

[2]. Ravikant G. Biradar, Abhishek Chatterje, Prabhakar Mishra, Koshy George"FPGA Implementation of a Multilayer Artificial Neural Network using System-onChipDesign Methodogoly". IEEE TRANSACTION 2015

[3]. Qiang Liu, *Member, IEEE,* Ming Gao, and Qijun Zhang, *Fellow, IEEE* "Knowledge-Based Neural Network Model for FPGA'' IEEE Transction On VerLarge Scale Integration (VLSI) System 2015 IEEE Transaction p.p 1063-8210

[4]. Mr Prashant D. Deotale Prof. Lalit Dole "Design of FPGA Based General Purpose Neural Network " ICICES2014

[5]. S.Hariprasath T.N.Prabakar " FPGA Implementation of Multilayer Feed Forward Neural Network Architecture Using VHDL"

[6]. Suhap Sahin, Yasar Becerikli, and Suleyman Yazici "Neural Network Implementation in Hardware UsingFPGAs" pp. 1105Springer-Verlag Berlin Heidelberg 2006

[7]. Rafid Ahmed Khalil Sa'ad Ahmed Al-Kazzaz "Digital Hardware Implementation of Artificial Neurons Models Using FPGA"Department of Electrical Engineering University of Mosul, Mosul, Iraq

[8]. Haitham Kareem Ali and Esraa Zeki Mohammed “Design Artificial Neural Network Using FPGA” IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.8, August 2010

[9]. Ayman Youssef, Karim Mohammed, Amin Nassar “Reconfigurable, Generic and programmable Feed Forward Neural-Network implementation in FPGA”2012 14th International Conference on Modelling and Simulation978-0-7695-4682-7/12 2012 IEEE DOI 10.1109/UKSim.2012.12

[10]. S. Haykin, "Neural Networks: a comprehensive foundation", 2nd Edition, Prentice Hall, 1999.

[11].S.Haykin, “*Neural Networks*”, Macmillan College Publishing Co. Inc.1994.

[12]Pravin Kshirsagar and Sudhir Akojwar, “Hybrid Heurictic Optimization for Benchmark Datasets”,IJCA,Vol 146-No.7,July 2016