Hardware Implementation of Backpropagation Neural Networks on Field programmable Gate Array (FPGA)

Rafid Ahmed Khalil
rafidamori@yahoo.com
University of Mosul, College of Engineering, Mosul, Iraq

Abstract

In this paper, a design method of neural networks based on VHDL hardware description language, and FPGA implementation is proposed. A design of a general neuron for topologies using backpropagation algorithm is described. The sigmoid nonlinear activation function is also implemented. The neuron is then used in the design and implementation of a neural network using Xilinx Spartan-3e FPGA. The simulation results obtained with Xilinx ISE 8.2i software. The results are analyzed in terms of operating frequency and chip utilization.

Key words : Artificial, Neural , Network, Backprobagation, FPGA,VHDL.

تنفيذ البنية المادية للشبكات العصبية على شريحة مصفوفة البوابات المبرمجة حقليا

رافد احمد خليل

كلية الهندسة ـ جامعة الموصل

الخلاصة

في هذا البحث، تم اقتراح طريقة لتصميم وتنفيذ الشبكات العصبية بوصف بنيتها المادية باستخدام لغة وصف الكيان المادي (VHDL) ، وتنفيذها على شريحة مصفوفة البوابات المبرمجة حقليا (FPGA). بدأ التصميم بخلية عصبية واحدة جعلت نواة لبناء هيكل وتنفيذ الشبكة العصبية متعددة الطبقات والخاصة بخوارزمية الانتشار العكسي BP. كذلك تم تنفيذ الكيان المادي لدالة التفعيل الغير خطية نوع سكمويد والمستخدمة في خوارزمية الانتشار العكسي BP . بعد ذلك تم تنفيذ تصميم البنية المادية الكاملة للشبكة العصبية الموصوفة بلغة وصف الكيان

المـادي (VHDL) علـى منصـة تطـوير البنيـة الماديـة الخاصـة بشـركة ®Xillinx. أن نتـائج المحاكـات والاختبـار للتصـميم أجريـت فـي البيئـة البرمجيـة Xilinx ISE 8.2i . أن نتـائج هذا البحث تم تحليلها من وجهة نظر تردد الاشتغال ونسبة استهلاك موارد الشريحة.

Received 29 May 2007 **Accepted 2 Sep. 2007**

1. Introduction

Artificial neural networks (ANN) have been used successfully in pattern recognition problems, function approximation, control, etc. Their processing capabilities are based on a parallel architecture [1]. There are different kinds of electronic implementations of ANN : digital, analog, hybrid, and each one has specific advantages and disadvantages depending on the type and configuration of the network, training method and application . For digital implementations of ANN there are different alternatives : custom design, digital signal processors, programmable logic. Among them, programmable logic offer low cost, powerful software development tools and true parallel implementations [2].

Field programmable gate arrays (FPGA) are a family of programmable logic devices based on an array of configurable logic blocks (CLB), which give a great flexibility in the development of digital ANNs [3].

The backpropagation algorithm [1], is one of the most useful algorithms of ANN training. In this paper, we present the neuron implementation for the in topologies that are suitable for this algorithm. The tanh sigmoid activation function is also implemented. The neuron is then used in a multilayer neural network.

For the implementation, VHDL language was used [4]. VHDL (Very high speed integrated circuit Hardware Description Language) is a hardware description language which simplifies the development of complex systems because it is possible to model and simulate a digital

system form a high level of abstraction and with important facilities for modular design.

The purpose of this work is to suggest and analyze several neuron implementations, show a way for the integration and control of the neurons within a neural network, and describe a way to implement a simple feedforward neural network trained by BP algorithm using XC3S500E Xilinx FPGA.

2. Neuron Implementation

The common mathematical model of a neuron is shown in Fig. (1) [1].

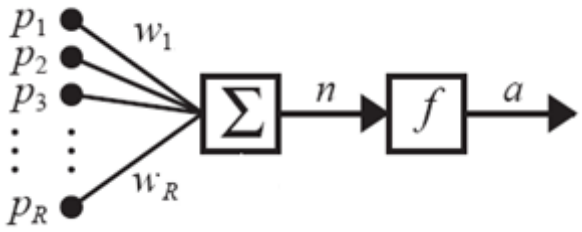


Fig. (1), Mathematical model of artificial neuron.

The neuron output can be written as:

$$a = f\left(\sum_{j=1}^{R} w_j \; p_j\right) \qquad \text{.....(1)}$$

where p_j are inputs, w_j are weight coefficients, f activation function, and a neuron output.

Two distinct neuron implementation were designed using 8-bit and 12-bit binary MAC (multiply accumulate) circuits. The sigmoid activation function is used for implemented neurons in hidden layer, and

linear activation function is used for output layer neuron. For all neurons (8-bit and 12-bit), the product of signed input (4-bit / 8-bit) and signed weight (4-bit) form a signed result (8-bit/ 12bit). These products value are accumulated into activation state. The final output value is obtained by applying the activation function. The weight coefficients are stored in a ROM within neurons.

Referring to Fig. (2), the MAC unit which accepts a serial processing of weights and parallel inputs pairs, each pair is multiplied together and a running total is recorded. An index control module controls the multiplexing order. Once all input pairs have been processed, the final sum is passed through the activation function to produce the neuron's output. The main advantage of serial processing is the small constant area required, regardless of topology, to implement one MAC and some routing for one input and one weight contained in the weight ROM module. The obvious disadvantage is the processing speed. If the network involves a large number of inputs, serial processing will suffer from slow processing.

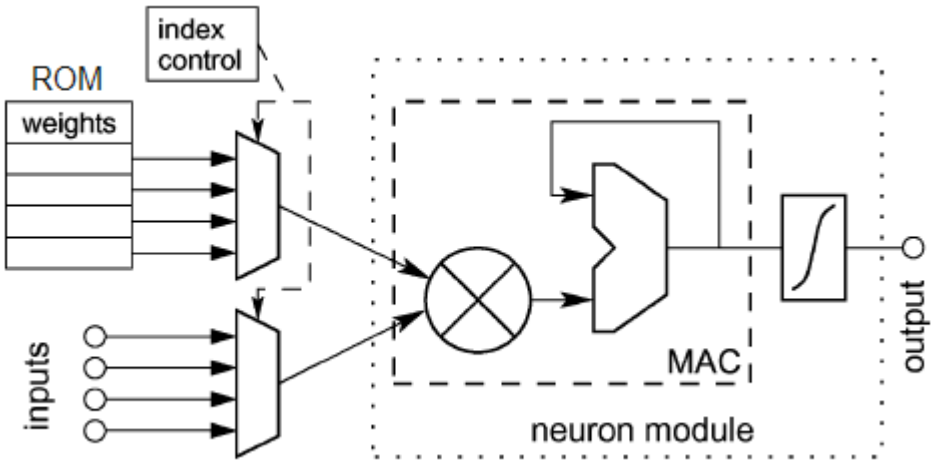


Fig. (2), Neuron structure—serial processing.

3. Sigmoid Activation Function Hardware Design

A very important part of neuron implementation is activation function hardware design. One of the most frequently used activation function in backpropagation neural networks applications is the hyperbolic tangent (tanh) sigmoid function (refered to as "tansig" in Matlab), and is given as:

$$f(n)=\frac{e^{n}-e^{-n}}{e^{n}+e^{-n}} \qquad \text{.....(2)}$$

This function is not suitable for direct digital implementation as it consists of an infinite exponential series. Many implementations use a lookup table for approximation. However the amount of hardware required for these lookup tables can be quite large especially if one required a reasonable approximation [5,6]. A simple second order nonlinear function exists which can be used as an approximation to a sigmoid function [7]. This nonlinear function can be implemented directly using digital techniques. The following equation is a second order nonlinear function which has a tanh-like transition between the upper and lower saturation regions :

$$f(n)=\begin{cases} 1 & for \quad L \le n \\ f'(n) & for \quad -L \prec n \prec L \\ -1 & for \quad n \le -L \end{cases} \qquad \text{.....(3)}$$

where L depends on the level of saturation of the function and $f'(n)$ is defined by

$$f'(n) = \begin{cases} n(\beta - \theta n) & for \quad 0 \le n \le L \\ n(\beta + \theta n) & for \quad -L \le z \le 0 \end{cases} \qquad \text{.....(4)}$$

where β and θ are parameter for setting the slop and gain. Fig.(3) shows the comparison between the sigmoid defined by equation (2) and the hardware approximation defined by equations (3 and 4).

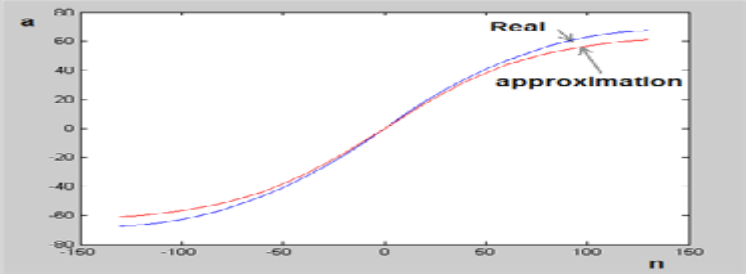


Fig.(3) Real tanh sigmoid activation function and hardware approximation.

For an 8-bit neurons, figures 4 and 5 show the time diagrams for implementing two neurons, one with approximated tanh sigmoid activation function and the other with linear activation function. The RTL (register transfer level) hardware circuits for implementing the two neurons are shown in figures 6 and 7, leading to different hardware complexity and different operating speeds.

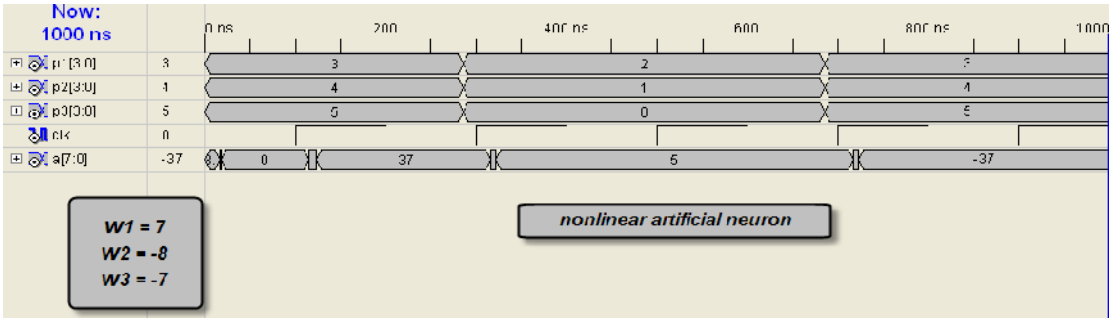


Fig.(4) Time diagram of implementing an 8-bit artificial neuron with approximated tanh sigmoid activation function.

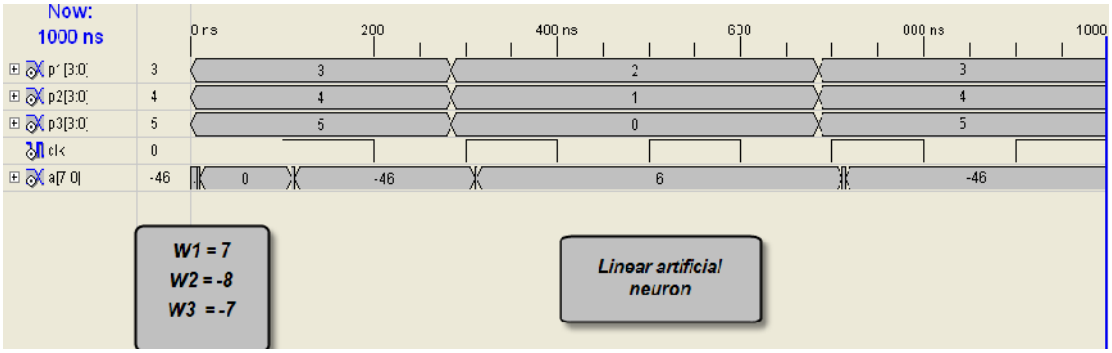


Fig.(5) Time diagram of implementing an 8-bit linear artificial neuron.

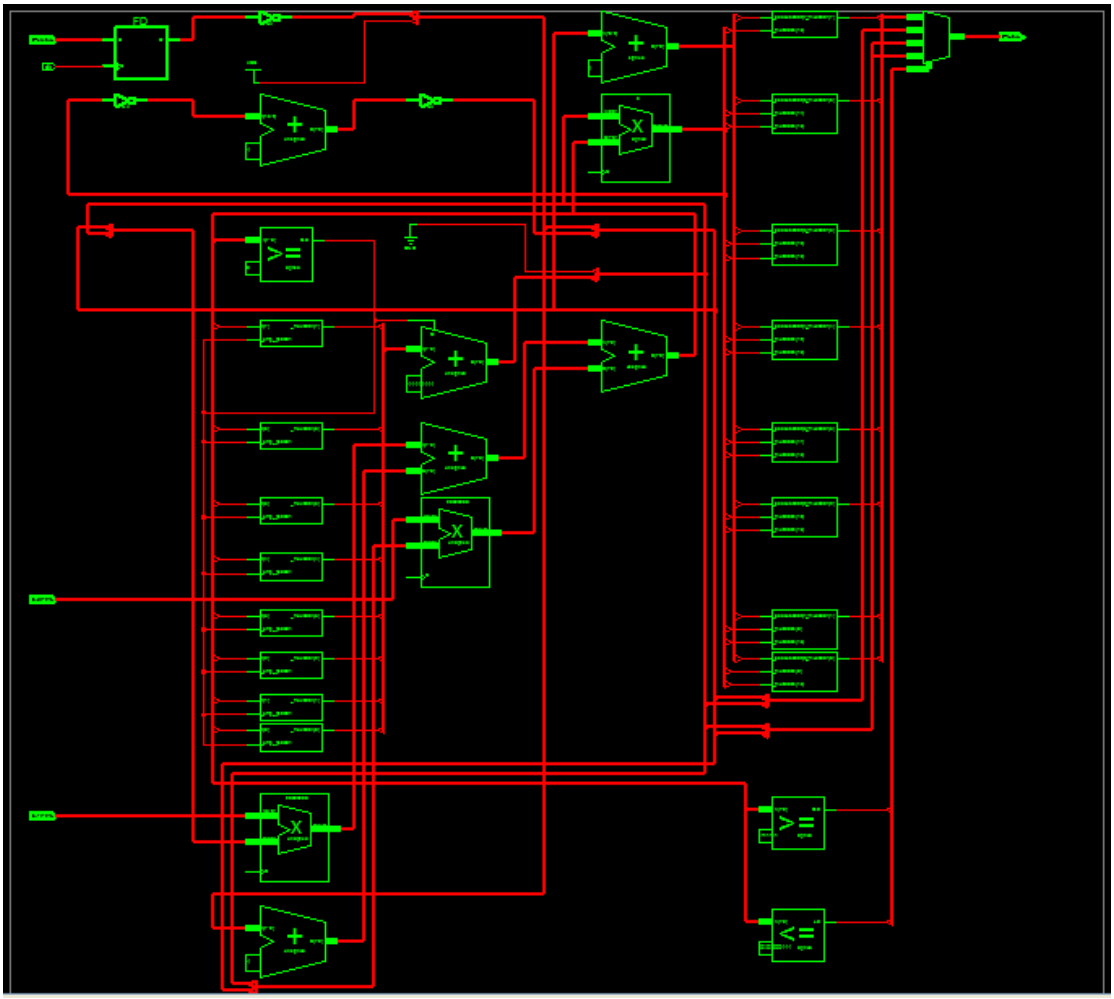


Fig.(6) RTL hardware schematic circuit for implementing tanh sigmoid artificial neuron.

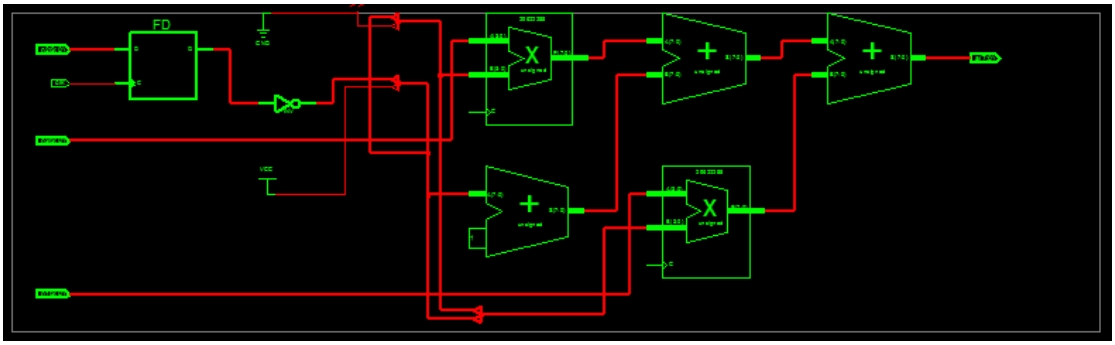


Fig.(7) RTL hardware schematic circuit for implementing linear artificial neuron.

4. comparison Results

Table (1) gives performance and resource use summary for the two implemented 8-bit neurons. As it can be seen, the linear neuron require very few hardware resource in comparison with tanh sigmoid nonlinear neuron. The operation speed in all cases gives a good results and shows the advantages of using FPGAs in neural realization.

Table (1) Comparative data for implemented 8-bit artificial neurons.

Neuron type	tanh sigmoid	Linear
Device utilization		
No.of Slices (4656)	50	18
No. of register	14	7
No. of 4 input LUTs (9312)	94	30
No. of bonded IOBs (232)	21	21
No. of Multiplier (20)	3	2
No. of GCLKs (24)	1	1
Time sumary		
Max path delay	24.8 nsec	12.7 nsec
Max operating frequency	40.3 MHz	78.7 MHz

Target device : xc3s500e fg320 -4
Software version : ISE 8.2i

5. Neural Network Simulation and Implementation Results

The architecture of feedforward neural network used in this work is 3-3-1 (input, hidden, output) layers. It is shown in Fig.(8).

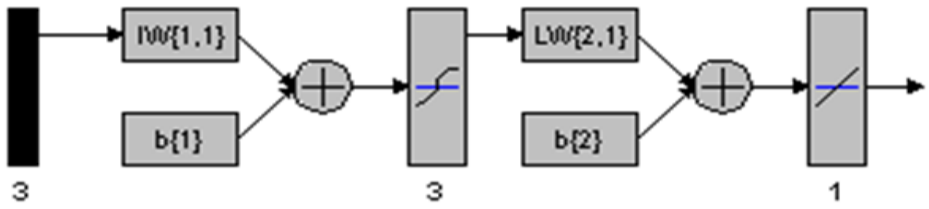
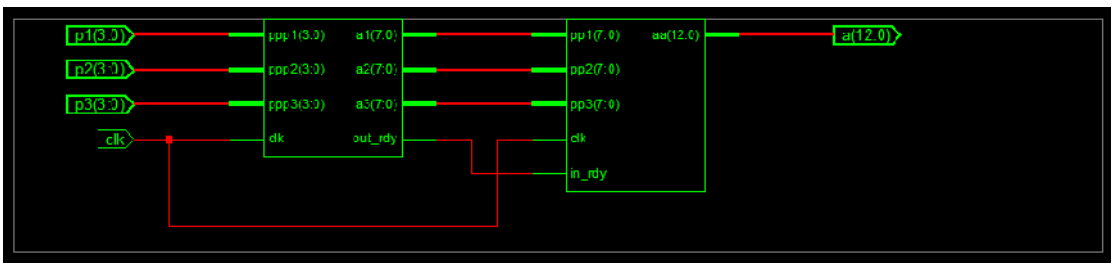


Fig.(8) Two layer feedforward (BP) neural network architecture of dimension

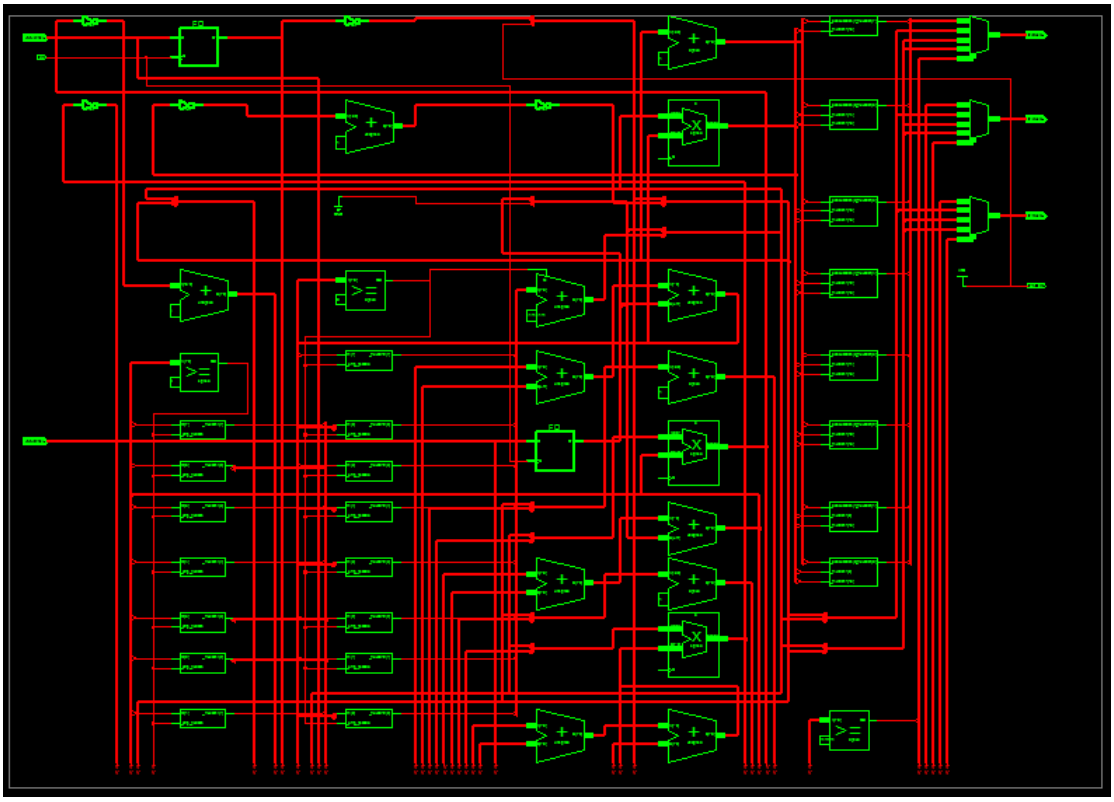
3-3-1 (referred to nntool matlab schematic notation).

The network is composed of three input , the hidden layer with three sigmoid neurons , and the output layer with single linear neuron. All neurons in the same layer are handled in parallelism. It fully uses the parallel, quick characteristic of the FPGA.

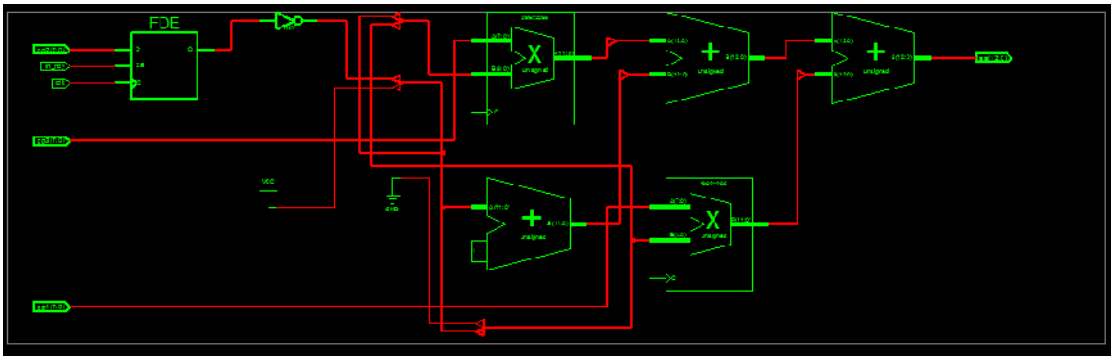
Considering the tradeoff area and speed in the design of the chip, the parallel inputs of every neuron from the previous layer are sent to the multiplication and accumulated to the activation function. The principle diagram of the top level and RTL level neural network design is shown in Fig.(9 a ,b, and c).



(a)



(b)



(c)

Fig.(9) (a) Top level diagram of implementing 3-3-1 feedforward neural network, (b) RTL level circuit diagram of implementing hidden layer, (c) RTL level circuit diagram of implementing output layer.

Where p_1, p_2, p_3 is the 4-bit input signals to hidden layer neurons. The 4-bit nine weights coefficients are stored in a ROM within neurons. The tanh sigmoid activation function of hidden neuron is implemented as VHDL package code according to kwan approximation [7]. The 8-bit outputs of hidden layer neurons a_1, a_2, a_3 are applied as input to output layer neuron. The output signal out_rdy is applied as start signal to output layer in_rdy . The clk signal drive both network layers. The three 4-bit weights coefficients of output layer are also stored in a ROM within output neuron . The overall network 12-bit output is a .

We adopt the ISE Xilinx foundation 8.2i software. The synthesized result is as follows:

-
```
Device utilization summary:
---------------------------

Selected Device : 3s500efg320-4

 Number of Slices:                      160  out of   4656     3%
 Number of Slice Flip Flops:             22  out of   9312     0%
 Number of 4 input LUTs:                310  out of   9312     3%
 Number of IOs:                          26
 Number of bonded IOBs:                  26  out of    232    11%
 Number of MULT18X18SIOs:                 5  out of     20    25%
 Number of GCLKs:                         1  out of     24     4%
```

-
```
Timing Summary:
---------------
Speed Grade: -4

   Minimum period: 25.133ns (Maximum Frequency: 39.788MHz)
   Minimum input arrival time before clock: 2.403ns
   Maximum output required time after clock: 14.248ns
```

To validate the performance of the neural network, we establish the test-bench considering the actual situation of the neural network operation. The test-bench adopting three type of input signal vectors, and the weight coefficient of hidden and output layers are stored in ROMs. The simulative results (time diagram) of the hidden layer and overall neural network are shown in figures 10 and 11 respectively.

As a result of synthesis and implementation of multilayer feedforword neural network on a Xilinx xc3s500 FPGA device. The device utilization summary, and timing summary gives a good results and shows the advantages of using FPGAs in neural realization.

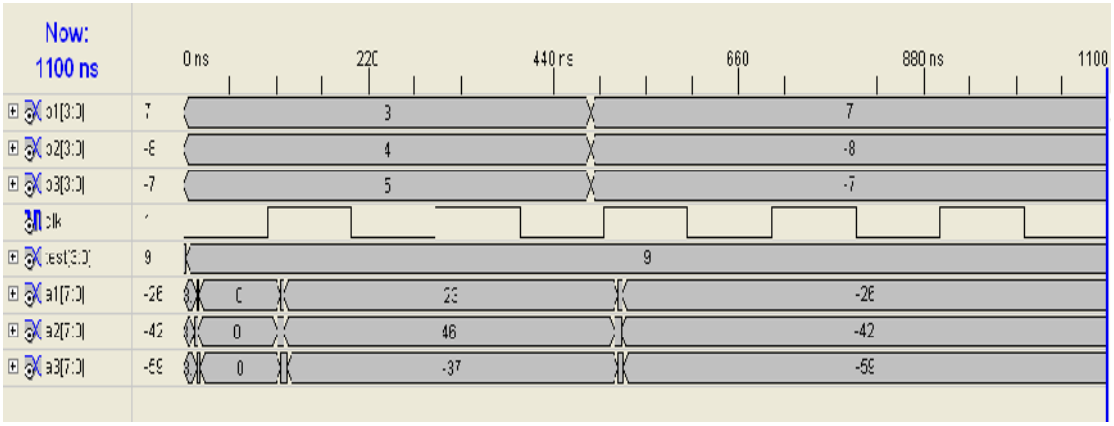


Fig.(10) Time diagram for (3-3) single layer tanh sigmoid neural network (assumed has identical performance to hidden layer).

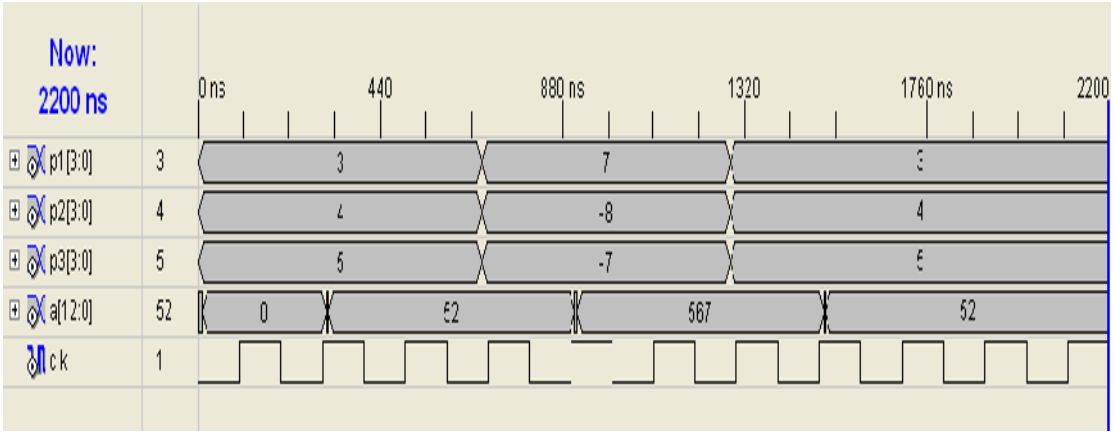


Fig.(11) Time diagram for (3-3-1) feedforward BP neural network.

6. Conclusions

Construction solutions for implementation of neural networks using FPGAs are described. The main purpose of this research was to design

and implement single neuron in the domain of speed and hardware complexity, and to suggest a solution for connecting neurons into a multilayer feedforward BP neural network. An important part of this work was the hardware implementation for the approximation of sigmoid activation function.

Since the more advanced families of FPGAs can contain more than 100,000 CLB (configurable logic block) [8], then it is clear that we can implement a network with an interesting number of neurons working in parallel in just a single chip. On the other hand, using hardware description, such as VHDL, represent a very practical option when dealing with complex systems. Finally, we can say the FPGAs constitute a very powerful option for implementing ANNs since we can really exploit their parallel processing capabilities.

References

1. M.Hagan , H . Demuth , M. Beele , " Neural Network Design" , University of Colorado Bookstore, 2002, ISBN : 0- 9717321- 0-8.
2. R. Omondi, C. Rajapakse," FPGA Implementation of Neural Networks", Springer U.S., 2006,ISBN 10-0-387-28485-0.
3. O. Maischberger ,v. Salapura , " A Fast FPGA Implementation of a General Purpose Neuron ", Technical University , Institute of in formatik , Austria , 2006.
4. D. L. Berry, "VHDL programming by examples", McGraw-Hill, fourth edition, 2002.
5. Pavlitov K., Mancler O., " FPGA Implementation of Artificial Neurons" , Electronics, No.9 September , 2004 , pp . 22-24 .
6. J. Blake , L. McDaid , " Using Xilinx FPGAs to Implement Neural Networks and Fuzzy Systems" , Faculty of Eng . , University . of Ulster , Magel college , Northland Rd . Derry , 2005.
7. Kwan , H.K. , " simple sigmoid . like activation function suitable for digital hardware implementation" , Electronic Letters , V.28 , July , 1992 , pp. 1379 – 1380 .
8. Xilinx , XST User Guide , Xilinx Inc . 2003.

The work was carried out at the college of Engg. University of Mosul