American Journal of Applied Sciences 11 (5): 782-788, 2014
ISSN: 1546-9239

doi:10.3844/ajassp.2014.782.788 Published Online 11 (5) 2014 (http://www.thescipub.com/ajas.toc)

VHDL DESIGN AND HARDWARE REALIZATION OF HYBRIDARTIFICIAL INTELLIGENCE ARCHITECTURE

[1]Rajeswaran Nagalingam, [2]Tenneti Madhu and [3]Munagala Suryakalavathi

[1]Department of ECE, SNS College of Technolgoy, Coimbatore, Tamilnadu, India
[2]Principal,Swarnandhra Institute of Engg. and Tech., Narasapur Andhrapradesh, India
[3]Department of EEE, Jawaharlal Nehru Technological University, Hyderabad Andhrapradesh, India

Received 2013-10-19; Revised 2013-11-11; Accepted 2014-02-23

ABSTRACT

Evolutionary Algorithms (EA) use Genetic Algorithm (GA) in many optimization problems to efficiently compute the function value in less time. In this study the weight optimization of the Artificial Neural Network (ANN), using the Back Propagation Network (BPN), is tested and presented with GA. The combined architecture of Neuro-Genetic (Hybrid Artificial Intelligence) approach is proposed and simulated results are provided along with device Utilization, Simulation time, Timing analysis and power analysis by using very high speed integrated circuits Hardware Description Language (HDL).

Keywords: ANN, Evolutionary Algorithm, GA, Hybrid AI, VHDL

1. INTRODUCTION

Charles Darwin proposed the “Theory of Evolution” in the year 1858. The Evolutionary Algorithms (EA) are broadly classified into three categories. They are Evolutionary Strategies (ES), Evolutionary Programming (EP) and Genetic Algorithms (GA) (Palmes *et al*., 2005). Ingo Rechenberg and Hans-Paul Schwefel (1960s and early 1970s) solved complex engineering problems through artificial evolution strategies using optimization method. John Holland is the initiator for the development of Genetic algorithms in 1970s (Mitchell, 1998). Further, the motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform “intelligent” tasks similar to those performed by the human brain. In 1943 the first neuron model was developed by the McCulloch and Pitts. Afterwards Evolutionary Computation (EC) and Neural Network technology impressed the interest of many researchers (Bose, 2007). A neural network is a powerful data modeling tool that is able to capture and represent the complex input/output relation-ships (Haykin, 1999). The development of back propagation algorithm in 1986 by Rumelhart and McClelland paved the way to research and development in the field of neuro computation till now. Most of the other neural network structures represent models for “thinking” that are still being evolved in the laboratories. However, Genetic algorithms are a class of optimization procedures which are good at exploring a large and complex space in an intelligent way to find values close to the global optimum. Hence, they are well suited to the problem of training feed for-ward networks (Montana and Davis, 1989). The power and usefulness of artificial neural networks has been demonstrated in several applications including speech synthesis, diagnostic problems, medicine, business and finance, robotic control, signal processing, computer vision and many other problems that fall under the category of pattern recognition (Alsmadi *et al*., 2011).

The main advantage of evolutionary computation has inspired new resources for optimization problem solving, such as the optimal design of neural networks and fuzzy systems (Juang, 2004). Field Programmable Gate Array (FPGA) are becoming increasingly popular for prototyping and designing complex hardware systems. The structure of FPGA can be described as an array of blocks connected together via programmable interconnections (Blake *et al*., 1997; Sahin *et al*., 2006). The relatively low cost and easiness

Corresponding Author: Rajeswaran Nagalingam, Department of ECE, SNS College of Technolgoy, Coimbatore, Tamilnadu, India

of implementation and reprogramming of FPGA's in comparison with the custom VLSI technology offer attractive features for the designer (Botros and Abdul-Arir, 1994; Ameur *et al*., 2012; Mahyuddin *et al*., 2009) Here in this study, the neuro-Genetic system is designed and simulated by using VHDL.

2. NEURO-GENETIC APPROACH

The traditional problem solving approach analyses the task and then derives a suitable algorithm. If successful, the result is immediately available. Neural networks can solve problems which are difficult to describe in analytical manner. But prior to usage, the network must be trained (Lange, 2005). A feed forward Back Propagation neural network is used to estimate the weights. The optimum selection of weights reduces the search-pace of the Genetic Algorithm (GA) to improve its performance (Kannaiah *et al*., 2011; Ditthakit and Chinnarasri, 2011). **Figure 1**, the evaluation function or fitness function is used to determine the fitness value of each candidate in the population. The proposed combined architecture will be helpful to solve the hardware implementation of the Hybrid AI.

2.1 Training Algorithm of BPN

It involves four steps (Sivanandam and Deepa, 2006)

Step 1: Initial weight selection
Step 2: Feed forward network
Step 3: Error calculation
Step 4: Updating of the weight and biases

The parameters are

x: Input training vector (x_1,……………x_n)
t: Output target vector
Δ_k: Error at output unit y_k
Δ_j: Error at hidden unit z_j
α: Learning rate
v_{oj}: Bias on hidden unit j
zj: Hidden unit j
w_{ok}: Bias on output unit k
y_k: Output unit k Equation (1):

$$y_k = w_{ok} \sum_{j=1}^{n} z_j w_{jk} \tag{1}$$

2.2. Procedure for Genetic Algorithm

The important parameters used in genetic algorithm are crossover rate, mutation rate, population size, selection, encoding and crossover and mutation type.

The algorithm is as follows:

Step 1: Initial- Random selection of population
Step 2: Select parents from population
Step 3: Perform crossover on parents creating an offspring
Step 4: Mutation operation for the best Individuals (offspring)
Step 5: Determine the fitness
Step 6: Repeat the process for the selection of the best individual and discard the worst

2.3. Pseudo Code for Proposed Design

```
va0,va1,va2: In integer range 1 to 15;
std_logic_vector(7 downto 0);
y:        Inout integer range 1 to 15);
std_logic_vector(7 downto 0));
signal za,zb,zc: Integer range 1 to 15 ;
component lfsr12bit is
clk, rst : In std_logic;
lfsr_reg : Inout std_logic_vector(11 downto 0));
end component; GA
Rand: In std_logic_vector(3 downto 0);
Wheel: In          std_logic_vector(3 downto 0);
GA_STORE1: Inout std_logic_vector (5 downto 0);
GA_STORE2: Inout std_logic_vector (5 downto 0);

----processing of 1st layer -- identity activation function
for hidden layer nodes---for number in 1 to 50 loop
        if (t/ = y ) then
        za0<=((xa*va0)+(xb*vb0)+(xc*vc0)+v0);
        zb0<=((xa*va1)+(xb*vb1)+(xc*vc1)+v1);
--identity activation function for hidden layer nodes
        if(clk'event and clk = '1') then
        if(reset = '1') then
        za<= 1;
-- calculation of error information
        if(clk'event and clk ='1') then
        if(reset = '1') then
        err_hid_za<= 1;
        if(clk'event and clk ='1') then
        if(reset = '1') then
        va0_new<= 1;
---calculation of weight compensation value
        if(clk'event and clk ='1') then
        if(reset ='1') then
        del_w0<= 1;
---calculation of new weight
        if(clk'event and clk ='1') then
        if(reset = '1') then
        wza_new<= 1;
end process;
```

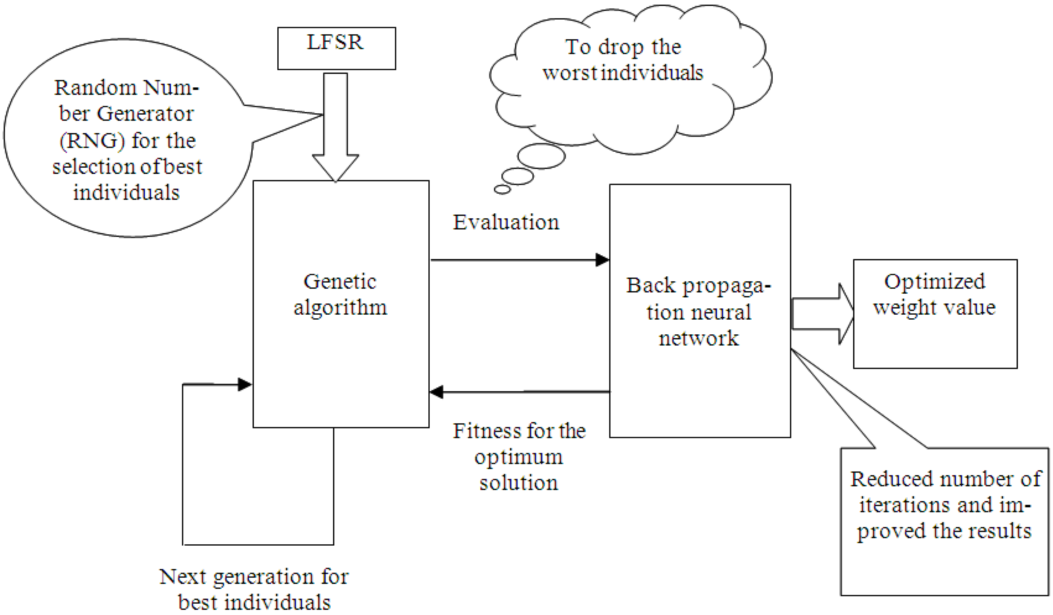

Fig.1. Proposed neuro-genetic approach

3. RESULTS

GA with BPN is tested by using the hardware unit xc3s500e-4-pq208. The simulation result, device utilization and timing summary are tabulated in **Table 1 to 3** respectively. The total number of Pins in the device are 208 and used pins in the proposed design are 49 only. Net skew is the difference between the minimum and maximum routing only delays for the net. Clock skew is the difference between the minimum and maximum path delays which include logic delays. The obtained clock report is presented in **Fig. 4.** Here the combined architecture (**Fig. 2**) used the roulette wheel selection method for fitness calculations. Based on the application, different methods can be used for fitness calculations. The total memory usage is 167380 kilobytes. The Total time to complete the simulation is 6.189 ns (4.567 ns logic, 1.622 ns route) (73.8% logic, 26.2% route). Total REAL time to Xst completion: 11.00 sec and Total CPU time to Xst completion: 11.48 sec. The output of the neuro-genetic is given in **Fig. 3**. The selected device consumed voltage, thermal and power information's are given in the screen shots (**Fig. 5-7**). It is evident that the utilization of hardware and efficiency to reduce power consumption in various factors of the proposed design are achieved.

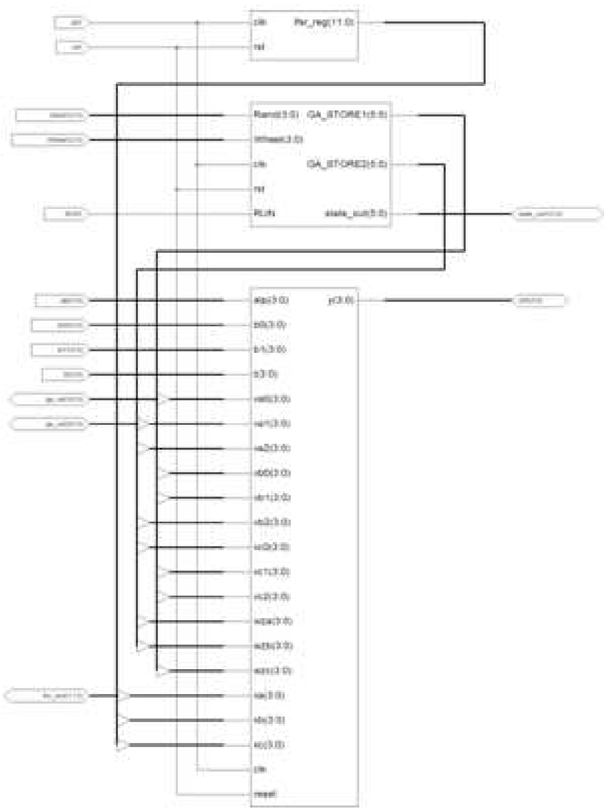

Fig. 2. FPGA implementation of Hybrid AI

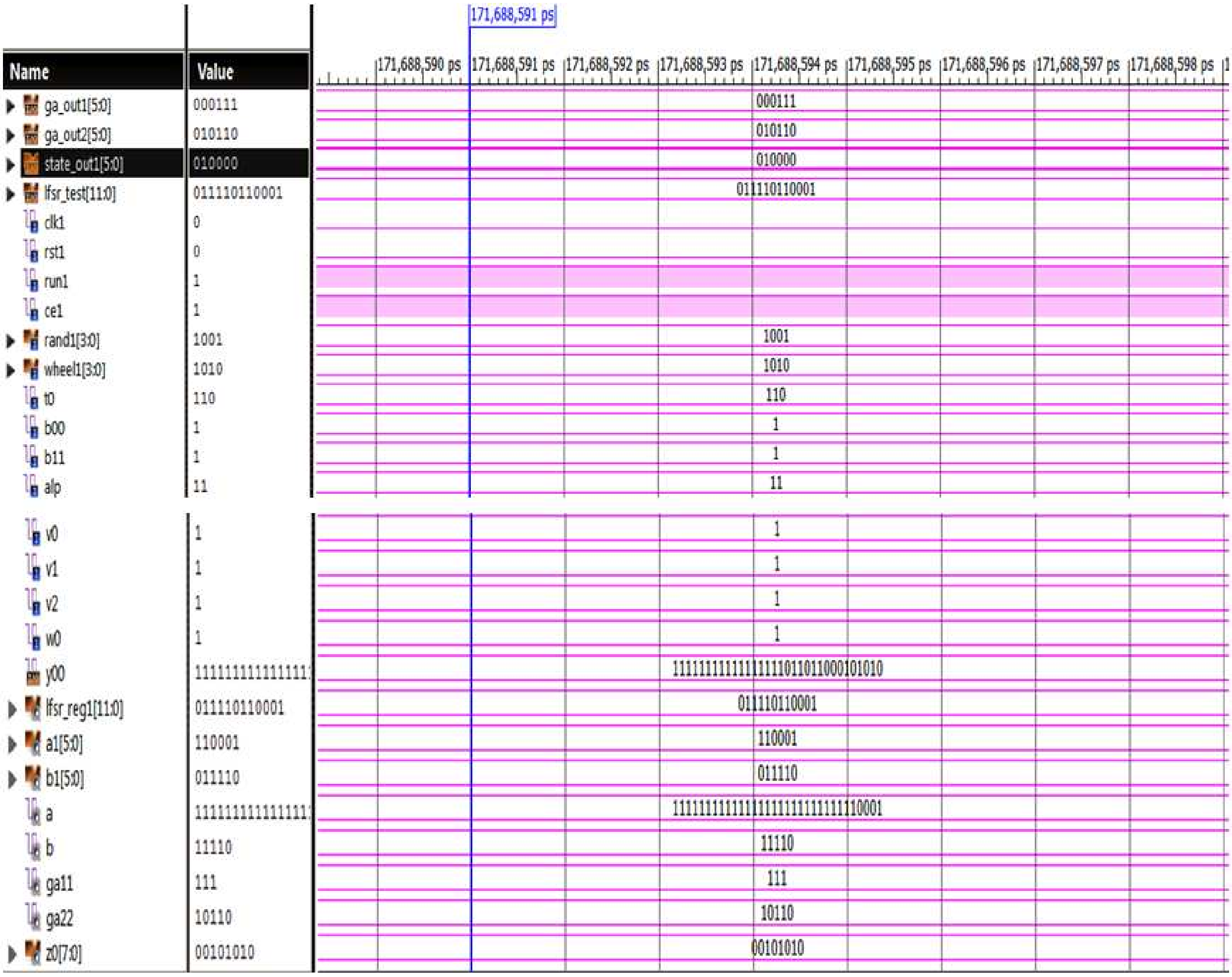

Fig. 3. Output waveform of proposed design

Clock Net	Resource	Locked	Fanout	Net Skew(ns)	Max Delay(ns)
state_out1_1_OBUF	BUFGMUX_X1Y11	No	11	0.025	0.157
clk1_BUFGP	BUFGMUX_X2Y10	No	34	0.029	0.153
x2/ov2	Local		6	0.071	1.297
x2/ov4	Local		15	0.758	2.559
x2/ov1	Local		6	0.045	2.000
x3/y0_not0001	Local		16	0.488	2.408
x2/ov3	Local		5	0.123	2.010

Fig. 4. Clock report

Name	Power (W)	Range	Voltage	Icc (A)	Iccq (A)
Vccint	0.031	1.14 to 1.26	1.20	0.000	0.026
Vccaux	0.045		2.5	0.000	0.018
Vcco25	0.005		2.5	0.000	0.002

Fig. 5. Voltage source information

Name	Value	Range
Ambient Temp (degrees C)	25.0	-20.0 to 85.0
ThetaJA (degrees C/W)	36	
Use custom ThetaJA ?	Yes	
Custom ThetaJA (degrees C/W)		0.0 to 20.0
Airflow (LFM)	NA	0 to 750
Max Ambient (degrees C)	82.1	
Junction Temp (degrees C)	27.9	

Fig. 6. Thermal information

Name	Value	Range
FF Toggle Rate (%)	12.5	0.0 to 100.0
I/O Toggle Rate (%)	12.5	0.0 to 100.0
Output Load (pF)	5.0	0.0 to 1000000.0
I/O Enable Rate (%)	100.0	0.0 to 100.0
BRAM Write Rate (%)	50.0	0.0 to 100.0
BRAM Enable Rate (%)	25.0	0.0 to 100.0
DSP Toggle Rate (%)	12.5	0.0 to 100.0
Part	3s500epq208-4	
Package	pq208	
Grade	Commercial	
Process	Typical	

Fig. 7. Power analyzer summary

Table 1. Device utilization summary

Logic utilization	Used	Available	Utilization (%)
Total number slice registers	119	9312	1
Number used as flip flops	43		
Number used as latches	76		
Number of 4 input LUT's`	96	9312	1
Number of occupied slices	107	4656	2
Number of bonded IOB's	49	158	31
Number of mulT18X18SIOs	12	20	60
Number of BUFGMUXs	2	24	8

Table 2. Timing summary

Parameters	Time (ns)
Minimum period	4.552
Minimum input arrival time before clock	5.711
Maximum output required time after clock	6.189
Maximum combinational path delay	No path found

Table 3. Simulation parameters

Parameters	Name of the signals	Value
RUN1	Processing signal	1
clk1	Input	Rising edge
rst1	Input	1
ce1	Processing signal	1
Rand1	Fitness	4 Bits
Wheel1	Fitness	4 Bits
GA_STORE1	Inout	6 Bits
GA_STORE2	Inout	6 Bits
state_out	Inout	6 Bits
t0	Target output	Integer range 1 to 15
b00	Bias1	Integer range 1 to 15
b11	Bias2	Integer range 1 to 15
alp	Learning rate	Integer range 1 to 15
v0	Weight of input layer	Integer range 1 to 15
v1	Weight of input layer	Integer range 1 to 15
v2	Weight of input layer	Integer range 1 to 15
w0	Weight of output layer	Integer range 1 to 15
lfsr	Linear feedback shift register	12 Bits
y00	Actual output	Integer range 1 to 15

4. CONCLUSION

The proposed method of neuro-genetic hardware architecture design is one of the successful implementation methods of Hybrid AI in FPGA. The device utilization, clock report, power summary, voltage source information, thermal Information and power analyzers show that the proposed method is achieving the required speed and efficiency. In future the proposed method will be used for different real time applications.

5. REFERENCES

Alsmadi, M.K., K.B. Omar and S.A.M. Noah, 2011. Fish classification based on robust features extraction from color signature using back-propagation classifier. J. Comput. Sci., 7: 52-58. DOI: 10.3844/jcssp.2011.52.58

Ameur, M.S.B., A. Sakly and A. Mtibaa, 2012. Implementation of genetic algorithms using FPGA technology. Proceedings of the Annual FPGA Conference, Sep. 4-4, New York, DOI: 10.1145/2451636.2451639

Blake, J.J., L.P. Maguire, T.M. McGinnity and L.J. MCDaid, 1997. Using xilinx FPGAs to implement neural networks and fuzzy systems. Proceedings of the Colloquium on Neural and Fuzzy Systems: Design, Hardware and Applications, May 9-9, IEEE Xplore Press, London, pp: 1-4. DOI: 10.1049/ic:19970730

Bose, B.K., 2007. Neural network applications in power electronics and motor drives-an introduction and perspective. IEEE Trans. Indust. Electron., 54: 14-33. DOI: 10.1109/TIE.2006.888683

Botros, N.M. and M. Abdul-Arir, 1994. Hardware implementation of an artificial neural network using. Field Programmable Gate Arrays (FPGA's). IEEE Trans. Indust. Electron., 41: 665-667. DOI: 10.1109/41.334585

Ditthakit, P. and C. Chinnarasri, 2011. Estimation of pan evaporation coefficient using neuro-genetic approach. Am. J. Environ. Sci., 7: 397-401. DOI: 10.3844/ajessp.2011.397.401

Haykin, S.S., 1999. Neural Networks: A Comprehensive foundation. 2th Edn., Prentice Hall International, Upper Saddle River, ISBN-10: 0139083855. pp: 842.

Juang, C.F., 2004. A hybrid of genetic algorithm and particle swarm optimization for recurrent network design, IEEE Trans. Syst, Man Cybern. B Cybern., 34: 997-1006. DOI: 10.1109/TSMCB.2003.818557

Kannaiah, S.K., J. Thangavel and D.P. Kothari, 2011. A genetic algorithm based multi objective service restoration in distribution systems. J. Comput. Sci., 7: 448-453. DOI: 10.3844/jcssp.2011.448.453

Lange, R., 2005. Design of a generic neural network fpga-implementation. Proceedings of the CHEMNITZ University of Technology, Faculty of Electrical Engineering and Information Technology, Professorship of Circuit and Systems Design, (SD '05), pp: 1-143.

Mahyuddin, M.N., C.Z. Wei and M.R. Arshad, 2009. Neuro-fuzzy algorithm implemented in Altera's FPGA for mobile robot's obstacle avoidance mission. Proceedings of the IEEE Region 10th Conference TENCON, Jan. 23-26, IEEE Xplore Press, Singapore, pp: 1-6. DOI: 10.1109/TENCON.2009.5396012

Mitchell, M., 1998. Introduction to Genetic Algorithm. 1st Edn., MIT Press, Cambridge, Mass, ISBN-10: 0262631857. pp: 209.

Montana, D.J. and L. Davis, 1989. Training feedforward neural networks using genetic algorithms. Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI '89), San Francisco, pp:762-767.

Palmes, P.P., T. Hayasaka and S. Usui 2005. Mutation-based genetic neural network. IEEE Trans. Neural Netw.,16: 587-600. DOI: 10.1109/TNN.2005.844858

Sahin, S., Y. Becerikli and S. Yazici, 2006. Neural network implementation in hardware using FPGAs. Proceedings of the 13th International Conference, Oct. 3-6, Hong Kong, China, pp: 1105-1112. DOI: 10.1007/11893295_122

Sivanandam, S.N. and S.N. Deepa, 2006. Introduction to Neural Networks using Matlab 6.0. 1st Edn., Tata McGraw-Hill Education, New Delhi, ISBN-10: 0070591121. pp: 656.