Genetic Algorithm-based Neural Network For Accidents Diagnosis of Research Reactors On FPGA

Abdelfattah A. Ahmed[1]; Nwal Ahmed Alfishawy[2]; Mohamed A. Albrdin[1] and Imbaby I. Mahmoud[1]

[1]Atomic Energy Authority, Nuclear Research Center, Inshas, Egypt
[2]Minufiya University, Faculty of Electronic Engineering, Minuf, Egypt
fatt231153@gmail.com

Abstract: In a nuclear research reactors plant, a fault can occur in a few milliseconds, so locating the fault might be of utmost importance due to safety, and other reasons. Accordingly, there is an increasing demand for automated systems to diagnose such failures. Both Genetic algorithms and neural networks, which are inspired by computation in biological systems, are emerged as established techniques for optimization and learning. So, using Genetic Algorithm (GA)-Based Artificial Neural Network (ANN) to obtain the optimum construction of an Artificial Neural Network, and then implementing it on a field programmable gate array (FPGA) is very interesting due to its high performance and can easily be made parallel. This paper presents a hardware implementation of a neural network that had obtained from Genetic Algorithm (GA) using MATLAB's toolbox. The excellent hardware performance has been performed through the use of field programmable gate array (FPGA), on Xilinx chip, to diagnosis the Multi-Purpose Research Reactor of Egypt accidents patterns, to avoid the risk of occurrence of a nuclear accident. The neural network hardware model has been designed using Xilinx Software environment.
[Abdelfattah A. Ahmed; Nwal Ahmed Alfishawy; Mohamed A. Albrdin and Imbaby I. Mahmoud **Genetic Algorithm-based Neural Network For Accidents Diagnosis of Research Reactors On FPGA]** Journal of American Science 2012; 8(3):228-234]. (ISSN: 1545-1003). http://www.americanscience.org. 30

Keywords: Genetic algorithms (GA), Artificial Neural Networks (ANN), Nuclear Reactors, field programmable gate array (FPGA), Hardware Description Language (HDL).

1. Introduction

Locating a fault in a nuclear research reactors plant, which can occur in a few milliseconds, might be the utmost interest due to safety, and other reasons. The early detection of such plant's failure could prevent system malfunction or serious damage, which could also lead to disaster. Therefore, an intelligent fault detection and diagnosis system to deal with inaccurate information has also been greatly required [1-3].

Genetic Algorithms and Neural networks are two techniques for optimization and learning, so recently, there have been attempts to combine them. The genetic algorithm improves the chances of finding a global solution, due to its random nature. This process involves a large number of complex arithmetical operations. However, the software implementations don't have the desired performance [4-6].

An interesting method to increase the performance of the model is by using hardware implementations, where the hardware can execute the arithmetical operations much faster than software. Hardware implementations have other advantages such as reducing the cost, greater reliability in operation, reduced probability of equipment failure, better protection against and special operating condition [7].

This paper is organized as follows: After this introduction, Section 2 presents the Egyptian Second Nuclear Research Reactor. Section 3 explains the Genetic Algorithm-based Neural Network. Section 4 explains a single hardware neuron circuitry. Section 5 explains our complete Hardware Neural Network architecture. Section 6 explains the Hardware Implementation on FPGA. Section 7 shows the results and discussions of the proposed system performance. Section 8 is devoted to conclusion. Section 8 assigned to References, and finally, appendix A is devoted for all accidents wave forms.

2. Egyptian Second Nuclear Research Reactor

The Egyptian Second Nuclear Research Reactor, is a multipurpose (MPR), open pool type, 22MW power, light water cooled and moderated and with beryllium reflectors, which had been mainly designed for radioisotopes production for medical and industrial purposes, semiconductors production, activation analysis, neutron radiography and beam tube experiments, basic and applied research in reactor physics and training. The operation of the reactor is controlled and monitored using: the suspension and control system (SCS) and the Reactor Protection System (RPS). The SCS provide process information to the operator in charge allowing him to control the process systems evolution and reactor power. The RPS is basically a control system that generates the signals for the protective functions to be carried out by the safety systems. The RPS encloses all electrical and mechanical devices and Circuitry involved in generating those initiation signals associated with protective function carried out by the safety actuation

systems. The reactor protection system is based on intelligent units combined with hardwired voting protective logic's are placed in the instrumentation room. The detector and sensors are placed as close as possible to the variables that they supervise. The following accidents diagnosed: 1- Loss Of Flow Accident (LOFA), 2- Loss Of Power Supply (LOPS), 3- Loss Of Heat Sink (LOHS), 4- Small Loss OF Coolant Accident (SLOCA), 5- Medium Loss OF Coolant Accident (MLOCA), 6- Large Loss OF Coolant Accident (LLOCA), 7- Uncontrolled Slow Reactivity Insertion (USRI), 8- Uncontrolled Fast Reactivity Insertion (UFRI) and 9- Normal case[7].

3. Genetic Algorithm-based Neural Network

Genetic Algorithms and Neural networks are two techniques for optimization and learning, each with its own strengths and weaknesses, and have generally evolved along separate paths. For the diagnoses of nuclear reactor accidents, a computer program is developed using MATLAB environment for this purpose, a Genetic algorithm program was designed and employed to construct an artificial neural network. The system was designed, using Genetic algorithms (GAs), to construct an ANN (the optimum values of weights and biases that are required to construct such network). The data used in the application were collected by the aid of reactor operation crew and Safety Analysis Report (SAR) of the reactor, in addition to the Atomic Energy experts. The data sets are for the eight accidental cases (Classes) listed below; plus the normal operation case as shown in Figure (1). So the total cases, which we have, are nine. One of the best structures that were obtained is two layers ANN with correspondence values of weights and biases that are required to construct such network(Figure2). Figures (3&4) show weights and biases matrices that are required to construct layer1 layer2 respectively. One of the basic problems to implement neural network on reconfigurable hardware, is related to the neurons transfer functions.

The problem of representing the arithmetic operations using digital hardware is related to using some transfer function like the sigmoid function, (frequently used in the Multilayer Perceptron (MLP) model), is not easy to implement the design in the hardware environment. So, for our case, the sigmoid function has been substituted by the linear function during the run of program of GA that were used to obtain the optimum construction of such neural network [4-7].

4. Single Hardware Neuron Circuitry

An artificial neural network (ANN) is an information processing system that tries to simulate biological neural networks. The field of ANN was born in an attempt to overcome the limitations of the computer's ability to perform certain tasks. The neural network is not programmed but "trained" [7].

```
[0 0 0 0 1 1 1 1 1 1 1 1 1 1]
[0 0 0 0 0 1 1 1 1 1 1 1 1 1]
[1 1 0 1 1 0 0 1 1 1 1 1 1 1]
[1 1 1 1 1 1 1 0 1 1 0 1 1 1]
[1 1 1 1 1 1 1 0 0 1 0 0 1 1]
[0 0 0 0 1 1 1 0 0 0 1 0 1 1]
[1 1 1 1 1 1 1 1 1 1 1 0 0 0]
[1 1 0 0 1 1 1 1 1 1 1 0 0 0]
[1 1 1 1 1 1 1 1 1 1 1 1 1 1]
```

Figure (1): Sample of reactor accidents data patterns.

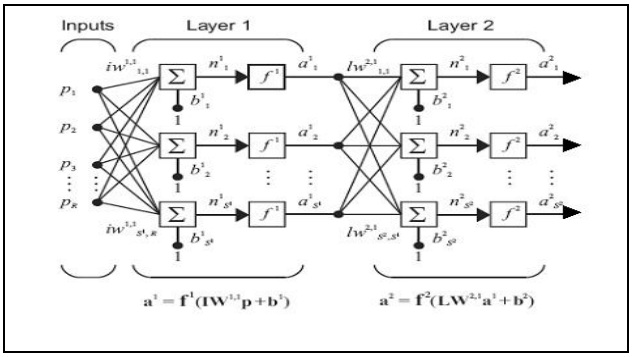

Figure (2): The output of two layers ANN structure.

0.671743	-0.35513	0.389441	0.616259	0.302567	-0.56619	-0.1325	-0.59441	0.350862	-1.00895
0.138234	-0.21709	-0.07338	-0.22197	0.268303	0.666455	1.151199	0.516719	-0.34027	0.115434
-0.55331	-0.90256	0.654735	-0.00592	-0.49488	0.788877	0.421017	0.466638	0.658911	0.221708
-0.17362	-0.50727	-0.98548	0.779943	-0.76442	-0.0305	-0.65107	0.722577	0.574992	0.058001
-0.32316	-0.74964	-0.07721	-0.08752	-0.19921	0.032543	-0.82106	0.199897	0.722563	0.913847
-0.58748	0.359816	0.227975	1.76068	0.322922	0.579638	0.081288	-0.10553	0.528519	-0.81042
-0.4354	-0.40606	0.059492	0.441864	0.129621	-0.66824	-0.33265	0.484437	-0.35653	-0.00428
0.091022	-0.2032	0.04947	-0.84382	0.52176	-0.68973	-0.76993	-0.21158	-0.472	-0.2261
-0.52835	1.008951	0.360588	-0.32831	0.663002	-0.39299	0.149951	-0.02365	0.904859	-0.15706
0.638713	0.172044	0.321576	0.774442	-0.18323	0.461319	0.172444	-0.54987	-0.51242	0.868027
-0.57516	-0.95342	0.034871	0.239162	0.309311	-0.48723	-0.40329	-0.16421	0.882939	-0.63421
0.658087	-0.01904	-0.15696	0.034275	-0.02869	-0.3625	-0.22912	0.133	-0.29314	-0.17731
0.24048	-0.57296	-0.20286	0.112262	-0.38607	0.474515	0.871335	0.498472	-0.0577	0.34246
-0.70995	0.73604	0.975565	0.513601	-0.39518	0.014475	-1.04897	-0.11151	-0.12001	-0.50271
-0.56796	-0.2308	-0.59528	0.075128	0.025215	0.217661	0.305784	0.315036	-0.67824	-0.68807
-0.44067	0.399412	-0.04543	-0.39277	0.79581	0.6578	0.617926	0.027264	-0.33026	-0.66042

Figure (3): Layer1 weights and biases matrix (16x10).

0.241942	-0.19631	0.333924	0.137175	-0.0124	-0.40334	-0.2798	0.106545	0.186543	
0.287052	-0.19428	0.021381	0.678396	-0.47925	0.299038	-0.21305	0.044092	-0.3108	
0.456773	0.557493	-1.07099	0.100792	-0.15013	0.31115	0.268079	-0.12595	0.493905	
0.10023	-0.34076	0.313971	-0.45207	0.573487	-0.65789	-0.33666	-0.01185	0.197631	
-0.03402	-0.17783	0.197313	0.59174	-0.4665	-0.65196	0.034814	0.523949	-0.25247	
0.489994	-0.42717	-0.1799	0.801007	-0.61729	0.784921	-0.28582	0.58967	-0.28177	
-1.04679	0.418243	-0.12505	-0.08194	0.225369	0.090697	0.04028	0.149455	-0.05086	
-0.05733	0.320575	0.080655	0.995438	-0.72909	-0.87798	0.518045	-0.36246	-0.08213	
0.00774	-0.18749	-0.05298	0.344698	-0.40748	0.514931	0.191211	-0.06111	-0.08526	
0.205339	0.278481	0.004393	-0.18188	0.443832	-0.87999	0.941548	-0.4985	-0.73389	
0.125281	0.83064	0.61907	-0.19462	0.742592	0.402752	1.003919	-0.75518	-1.52813	

Figure (4): Layer2 weights and biases matrix (16x10).

The design and all of our work are directed towards the implementation of a multilayer perceptron (MLP) in a modular way. Our modular design means that the network constructed initially with small

components and become as large as practically required to obtain the structure of the complex application. Figure (5) explain the expression to be implemented in the hardware, where Pi is the input signals, as depicted in figure (1), w_i the weights; f(x) is the activation function and α is the output, equation (1).

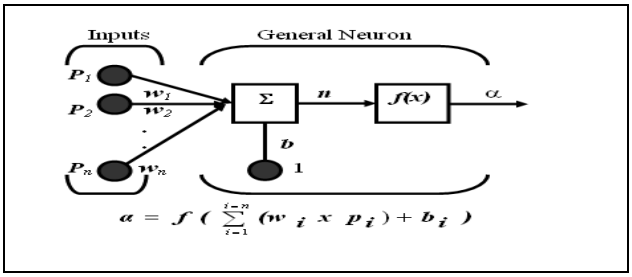

Figure (5): Single neuron module.

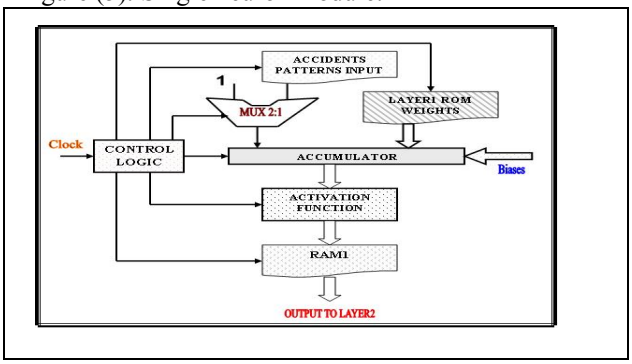

Figure (6): A block diagram of a complete hardware neuron of layer 1

Figure (6) depict the block diagram of complete hardware neuron that is used in layer1. The circuit does the algebraic equations of the electric model of the neuron, that is, the multiplication and sum required in the neuron's internal processing. Where the input patterns of the accidents are in binary form, we are Satisfied with the using of a multiplexer, as shown in figure (6), to enter the multiplication of the input values by the corresponding weights. In layer2 we were forced to use a multiplier to multiply the real values output from layer1 by the real values of the corresponding weights of layer2

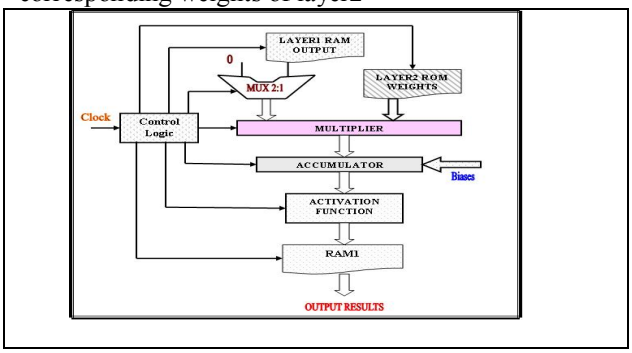

Figure (7): A block diagram of a complete hardware neuron of layer 2

5.Complete Hardware Neural Network architecture

The architecture of the complete hardware neural network is shown in figure (8), which includes the accidents input patterns, hidden layer (layer 1) and output layer (layer 2). Where, the design is modular and parametric so it can be easily expanded as long as device's resources allow it. So, when put together they need additional circuitry for synchronization and control. Private weights storage ROM has been assigned for each layer, where each neuron will access its memory storage area. One multiplier and one accumulator have been assigned for each layer. Data transfer between layers will stored in a private RAM for each layer, as shown in figure (8), to gather all computations of all neurons of each layer. Layer 1 output RAM is dedicated for storing layer 1 output and Layer 2 output RAM is dedicated for storing layer 2 output (so, the network output).

6. Hardware Implementation on FPGA

Neural networks, in general, work with floating-point numbers and working with floating-point numbers in hardware is difficult because the arithmetic operations are more complex than with integer numbers and the dedicated circuits for floating-point operations are more complex, slower, and occupy a larger chip area that integer [7,8]. So, one floating-point multiplier has been forced to use in layer2, to keep the computations precision and achieving an excellent results. But in layer1 the use of the multiplexer is enough because of the binary inputs of the accidents patterns.

The other problem of representing the arithmetic operations using digital hardware is the using of some transfer functions like the sigmoid function (frequently used in the MLP model). To overcome this problem and to make the design is easy to implement in the hardware environment, the construction of the ANN has been got using "Liner Transfer Function" when the GA-based ANN program was run. In this case, the sigmoid function has been substituted by a linear function is shown on Figure (9).

The hardware model of the neural network has been designed using Xilinx FPGA environment [12], where Xilinx offers competing products. Moreover, because Xilinx FPGA environment are standard parts that need only to be programmed, and no need to wait for prototypes or pay large nonrecurring engineering (NRE) costs. [ ]. One of its products is the Xilinx Spartan3A FPGAs, which are ideal for low-cost, high-volume applications and are targeted as replacements for fixed-logic gate arrays and Application Specific Standard Product (ASSP) products such as bus interface chip sets [7-12].

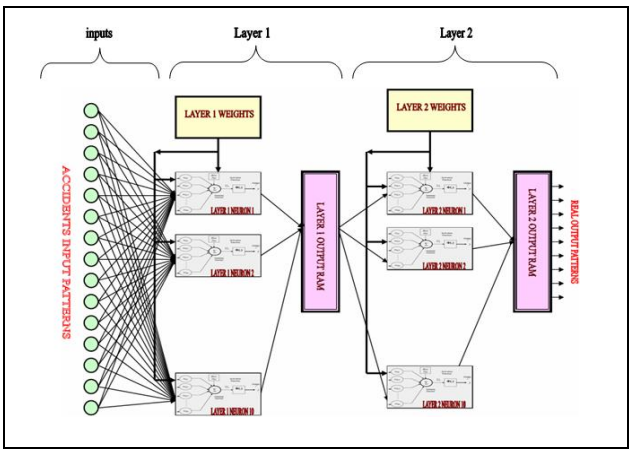

Figure (8): The block diagram of the complete hardware neural network.

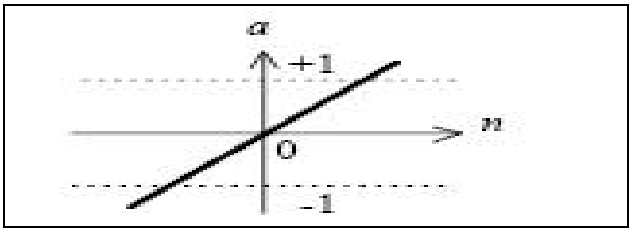

Figure (9): Linear Transfer Function (a = purelin (n))

The Xilinx design tools interface supports two basic flows within the Project Manager: HDL and Schematic. An HDL Flow project can contain VHDL, Verilog, or schematic top-level designs with underlying VHDL, Verilog, or schematic modules. VHDL is one of industry's standard languages used to describe digital systems. VHDL stands for VHSIC (Very High Speed Integrated Circuits) Hardware Description Language. VHDL has many features appropriate for describing the behavior of electronic components ranging from simple logic gates to complete microprocessors and custom chips [13-14].

Project Settings

Property Name	Value
Top-Level Source Type	HDL
Product Category	All
Family	Spartan3A and Spartan3AN
Device	XC3S700A
Package	FG484
Speed	-4
Synthesis Tool	XST (VHDL/Verilog)
Simulator	Modelsim-SE VHDL
Preferred Language	VHDL
Property Specification in Project File	Store all values
Manual Compile Order	☐
VHDL Source Analysis Standard	VHDL-93

Figure (10): Project information.

To generate hardware models from software algorithm some simple logic and arithmetic blocks such as: ROMs, RAMs, multipliers, adders, and logic gates have been designed. Initially, there are fifteen inputs that will multiply by the corresponding weight, and then added to the accumulator. The results of the addition are added with the assigned bias for each neuron. Finally, the transfer function delivers the neuron's output to layer1 RAM which stores the computation of each neuron. The connections between components of the complete neural network are dedicated hardware signal for data transfer. The signals (busses) distribute the computations to the assigned target. Finally, the result appears in the output pin (on the right side).

7. Results and discussion

The model needs less than 2 µs for processing the input values and presenting the result, as displayed from figure (11A). This is a very fast implementation comparing to the protection software system that needs from 18 ms to 24 ms as a response time. The neural network's hardware model will be integrated within the

Reactor Protection System (RPS) of the Multi-Purpose Research Reactor of Egypt (MPR) where, a valuable interesting result will obtain. The LOFA Accident simulation waveforms result of the proposed implementation is shown in Figure (11A). Figure (11B) and Figure (11C) show the desired output of LOFA accident and its real output respectively.

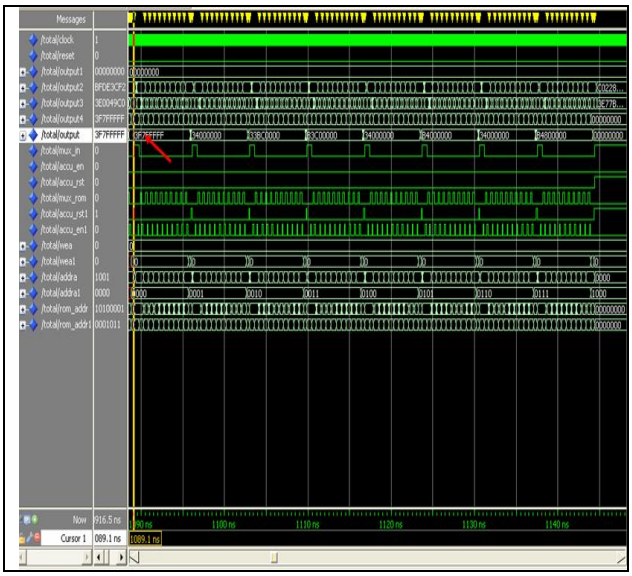

Figure (11A): The LOFA Accident Simulation Waveform

LOFA	1	0	0	0	0	0	0	0	0

Figure (11B): The desired output for Accident (LOFA**).**

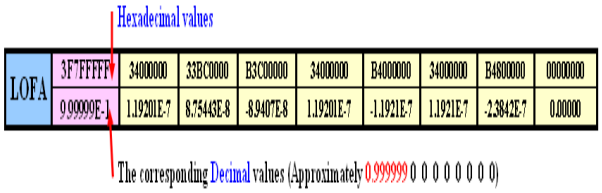

Figure (11C): The Hexa. Decimal and Decimal values for real output for Accident (LOFA)

7.1 Design Results Summary

This Project was designed to diagnosis and to predict the Multi-Purpose Research Reactor of Egypt accidents, to avoid the risk of occurrence of a nuclear accident. A sequential Hardware Neural Network is implemented using XC3S700A-4-FG484 device - Xilinx FPGA SPARTAN3A-3AN family. The implementation on FPGA provides the higher benefits of lower costs and higher results, where, FPGA can be reprogrammed for an unlimited number of times; they can be used in innovative designs where hardware is always in dynamic change, or where hardware must be adapted to different user applications requirements.

Table (1): Accident Simulation Waveform

```
Macro Statistics
# FSMs                 : 1
# ROMs                 : 2
 15x1-bit ROM          : 2
# Adders/Subtractors   : 7
 4-bit adder           : 3
 7-bit adder           : 2
 8-bit adder           : 2
# Registers            : 64
 Flip-Flops            : 64
# Latches              : 10
 1-bit latch           : 3
 4-bit latch           : 5
 7-bit latch           : 1
 8-bit latch           : 1
```

Table (2): Design Statistics

```
Design Statistics
=================
# IOs                   : 162
Cell Usage :
# BELS                  : 4059
#       GND             : 8
#       INV             : 12
#       LUT1            : 12
#       LUT2            : 482
#       LUT2_D          : 1
#       LUT3            : 690
#       LUT4            : 926
#       LUT4_D          : 4
#       MULT_AND        : 225
#       MUXCY           : 888
#       MUXF5           : 103
#       MUXF6           : 6
#       MUXF7           : 3
#       VCC             : 8
#       XORCY           : 691
# FlipFlops/Latches     : 230
#       FD              : 16
#       FDC             : 95
#       FDP             : 1
#       FDRS            : 80
#       LD              : 21
#       LD_1            : 17
# RAMS                  : 4
#       RAMB16BWE       : 4
# Clock Buffers         : 1
#       BUFGP           : 1
# IO Buffers            : 161
#       IBUF            : 1
#       OBUF            : 160
-----------------------------
```

8. Conclusion

This work proved the advantage of the hardware implementation of Neural Networks in a special hardware (FPGA), where the neural network is implemented on Xilinx chip. Where, we can say; the using of hardware description, such as VHDL, represents a very practical option when dealing with complex systems. Secondly, the using of FPGAs constitutes a very powerful option for implementing ANNs since we can really exploit their parallel processing capabilities in the nuclear reactors domain. The model needs less than 2 µs for processing the input values and presenting the results. This is a very fast implementation required for a critical field like the nuclear research reactors, comparing to the protection software system that needs from 18 ms to 24 ms as a response time. So, more precise classification accidents can be processed in a shorter period of time. The neural network's hardware model will be integrated within the Reactor Protection System (RPS) as a future work, where valuable interesting results will obtain.

Corresponding author
Abdelfattah A. Ahmed[1]
Atomic Energy Authority, Nuclear Research Center, Inshas, Egypt
fatt231153@gmail.com

References:

1. The International Atomic Energy Agency (IAEA) Safety Series, "Safety in the Utilization and Modification of Research Reactors", IAEA publications, STI/PUB/961, Vienna, Austria, December 1994, STI/PUB/961.
2. J. Korbicz, Z. Kowalczuk, J. M. Koscielny, W. Cholewa : "Fault diagnosis: models, artificial intelligence, applications", ISBN 3-540-40767-7, Springer-Verlag Berlin Heidelberg 2004.
3. B. Ch. Hwang, "Fault Detection and Diagnosis of a Nuclear Power Plant Using Artificial Neural Networks", Simon Fraser University, March 1993.
4. L. S. Admuthe and S. D. Apte, "Neuro – Genetic Cost Optimization Model: Application of Textile Spinning Process", International Journal of Computer Theory and Engineering, 1793-8201, Vol. 1, No. 4, October 2009.
5. A. Fiszelew, P. Britos, A. Ochoa, H. Merlino, E. Fernández, R. García-Martínez, "Finding Optimal Neural Network Architecture Using Genetic Algorithms", Software & Knowledge Engineering Center. University of Buenos Aires.Kb, Research in Computing Science 27, 2007.
6. A. M. Sadeq, A. A. Wahdan, H. M. K. Mahdi, "Genetic Algorithms and its Use with Backpropagation Network", Faculty of

Engineering, Ain Shams University; Vol. 35, No. 3, Sept 30, 2000.
7. S. Sh. Haggag, PhD Thesis, *"Design and FPGA-Implementation of Multilayer Neural Networks With On-chip Learning"*, Atomic Energy Authority, Egypt 2nd Research Reactor. PhD, Menufia University, 2008.
8. Amos R. Omondi and Jagath C. Rajapakse, "fpga implementations of neural networks", ISBN-13 978-0-387-28487-3 (e-book), Springer, P.O. Box 17, 3300 AA Dordrecht, The Netherlands, 2006.
9. kChristophe Bobda, "Introduction to Reconfigurable Computing - Architectures, Algorithms, and Applications", University of Kaiserslautern, Germany, ISBN 978-1-4020-6100-4 (e-book), Springer, 2007.
10. Uwe Meyer-Baese, "Digital Signal Processing with Field Programmable Gate Arrays", ISBN 978-3-540-72612-8 Springer Berlin Heidelberg New York, 2007.
11. Karen Parnell and Nick Mehta, "Programmable Logic Design Quick Start Handbook", Xilinx, Inc. PN 0402205 Rev. 4, 4/04, 2004.
12. Peter J. Ashenden, "The Designer's Guide to VHDL", Morgan Kaufmann Publishers, I

Appendix A: All Accidents Simulation Waveforms

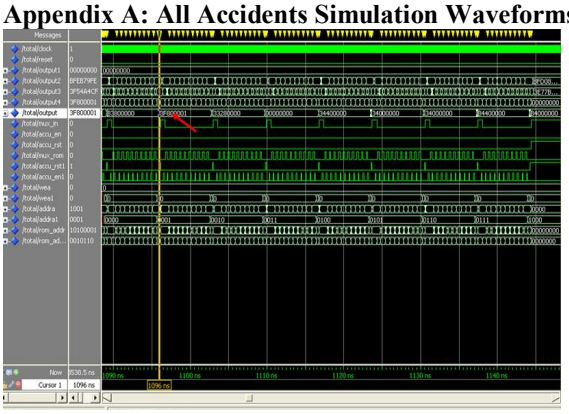

Figure (12): Loss Of Power Supply (LOPS) Accident Simulation Waveform

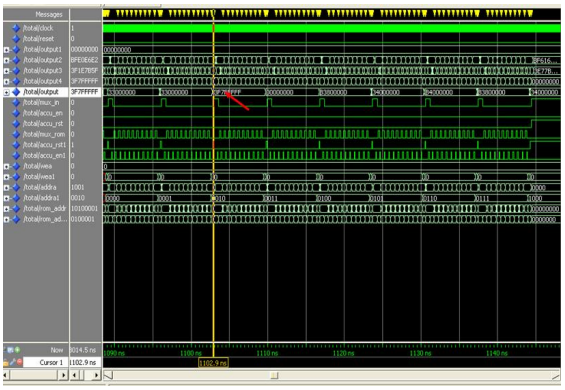

Figure (13): Loss Of Heat Sink (LOHS) Accident Simulation Waveform

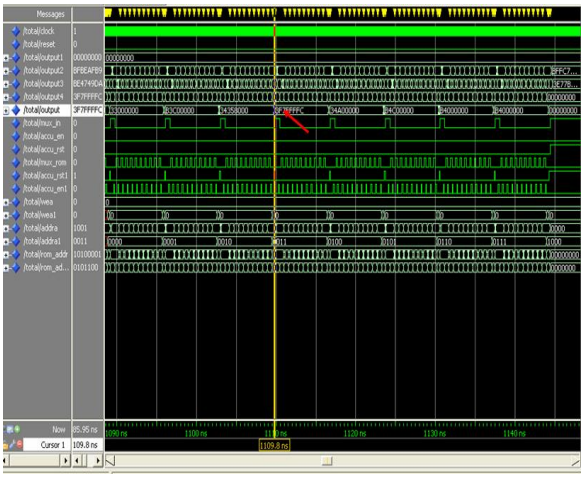

Figure (14): Small Loss OF Coolant Accident (SLOCA) Accident Simulation Waveform

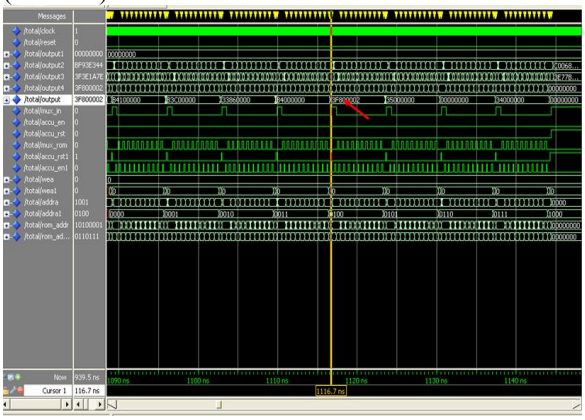

Figure (15): Medium Loss OF Coolant Accident (MLOCA) Accident Simulation Waveform

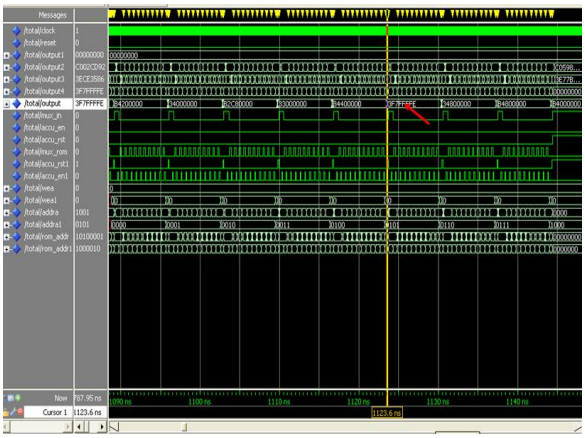

Figure (16): Large Loss OF Coolant Accident (LLOCA) Accident Simulation Waveform

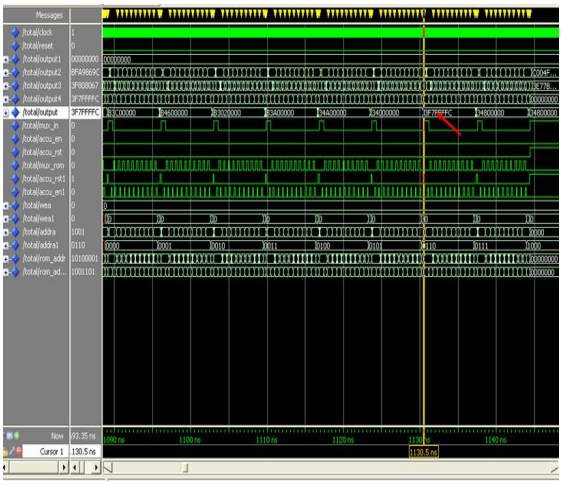

Figure (17): Uncontrolled Slow Reactivity Insertion (USRI) Accident Simulation Waveform

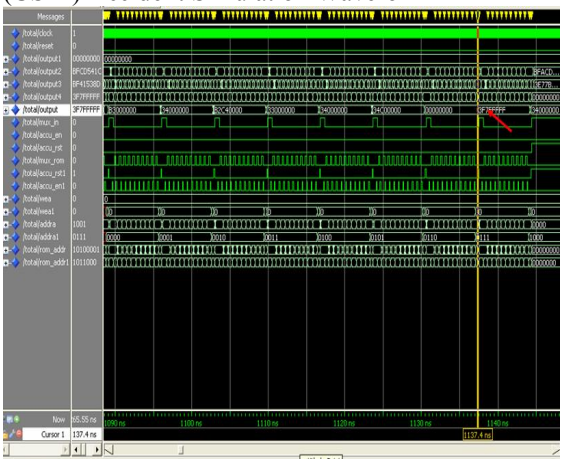

Figure (18): Uncontrolled Fast Reactivity Insertion (UFRI)

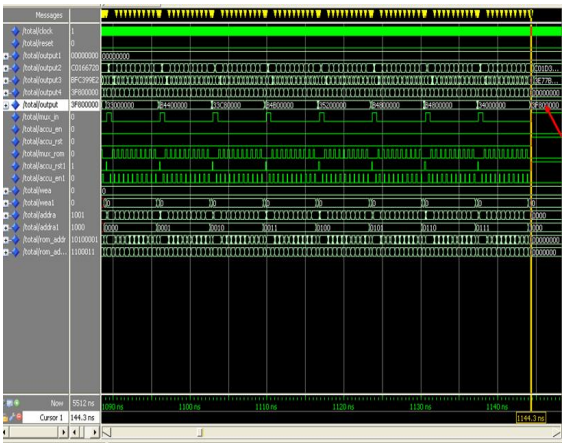

Figure (19): Normal case Accident Simulation Waveform

2/18/2012