Design and Implementation of Intelligent Controller Based on hybrid renewable energy source

Areeg Fadhil Hussein 1, Prof. Dr. Hanan A. R. Akkar 2

1Department of Electrical Engineering /University of Technology, Ministry of Education, Directorate of Vocational Education, Email: areeg.1987@gmail.com.
2Department of Electrical Engineering /University of Technology, Email: dr_hananuot@yahoo.com, 30084@uotechnology.edu.iq

Abstract. This paper presents the design and implementation of Artificial Neural Networks (ANN) that control a hybrid wind-solar system based on Field Programmable Gate Arrays. The controller designed to direct the output from the clean energy systems where the output might be from the wind, solar, the sum of the two systems, or from the battery charging system. The ANN designed and trained using MATLAB2019a, two training algorithms were explored Levenberg-Marquardt and Practical Swarm Optimization and the one with best training results have been chosen to be Co-Simulated on Xilinx Spartan 6 FPGA XC6SLX9FTG256 using Xilinx System Generator (XSG) with MATLAB R2012b Simulink then the FPGA card Hardware Implementation using ISE Design Suite 14.7 program.

1. INTRODUCTION
These Over the recent years, the use of renewable energy sources to reduce emissions, global warming and their environmental consequences has become a necessity for governments all over the world [1]. many works had been done to make the best use of renewable energy as an alternative source of energy, all of which began at the end of the previous century, as the level of pollution has risen due to the usage of fossil fuels and the near extinguishment of these sources of energy, so it became important to substitute fossil fuels with inexhaustible energy sources such as bioenergy, geothermal energy, solar energy, wind, etc.[2]. Solar energy influenced by daytime and weather conditions and seasons, as well as wind energy, fluctuates with seasonal changes, local terrain and losses in the wind turbine itself [3]. There are several ways to address the non-uniform response of solar and wind energy, such as Maximum Power Point Tracking (MPPT) algorithms used to harvest maximum power such that the device reaches its optimum efficiency [4][5]. MPPT algorithms are created using Artificial Neural Networks (ANN) by training the neurons on a data set using one of the supervised learning algorithms where the user provides the network with the desired output [6], one of the famous training algorithms is the backpropagation algorithm where it can be modified using many algorithms such as PSO [7]. Field Programmable Gate Array (FPGA) is the best microprocessor to implement the ANN with its high-speed, enhanced performance and easy-to-design high-speed ANN [8]. In 2016, Damodhar, Kumar presented an improved variable step-size P&O algorithm to implement Sinusoidal Pulse Width Modulation (SPWM) for a single-phase hybrid power filter generator for solar and wind grid applications. The algorithm then translated into Very high-speed integrated circuit Hardware Description Language (VHDL) which is implemented on the Spartan 3 FPGA. This algorithm acts as idiomatic controller in terms of tracking speed and reduce inconsistency of the output power [9]. In 2017 Aymen et al., proposed a Fuzzy Logic Controller to achieve the MPPT with fast time response for the solar-wind system, the controller was implemented on MATLAB and Xilinx FPGA blocks to estimate the timing response and the tracking efficiency, where the system with Xilinx System
Generator (XSG) settled quickly which achieved settling time about (0.4ms) [10]. In 2017 Akram et al., presented the scientific method for the Joint Capacity Optimization of Renewable Energy Sources by using Super Capacitance its value controlled by an optimization method to control the battery state of charge, they found that using the Super Capacitance provided an economic and reliable solution [11]. In 2018 Chishti, et al., developed concurrent generator based solar and wind microgrid battery system, they developed a prototype operate under variable conditions (different wind speed, disconnection and isolation on the solar panel) and study the system steady state as well as its dynamic performance. the results showed the effectiveness of their technique where all the load demands have been provided [12]. In 2019 Zarrad et al., proposed hybrid system consist of two sources of energy (solar and wind) and used an Artificial Neural Network controller to extract the maximum power at fixed atmospheric conditions and then optimize the system with respect to the real time qualifications so as to improve the performance of the system, ANN consists of two hidden layers one of twenty neurons and the second with single neuron all simulated by MATLAB and then implemented on XSG where the results confirmed that the system presents an acceptable execution real time performance and precision [13]. In 2020 Anand and Sundaram, suggested forecasting system predict that load management for solar and wind power systems is significant and then control the DC boost converter of the wind turbine and the solar system [14].

2. ARTIFICIAL NEURAL NETWORKS

The human brain is biological neural networks which perform the control and makes suitable decision related to the human body. The human brain has 100 billion neurons. The neurons connect between them by interconnections (synapse) and the human brain approximately has more than 125 trillion synapses [15]. The weights between the neurons are the essential elements for the long memory in ANNs [16]. The ANN contains set of information sources. Each input is multiplied by weight before it achieves the primary body of the cell, the weight represents the information about the inputs that is needed to tackle issue [17]. After multiplied input signals with associated weights, it enters the summing block thus generated net. The net is applied to activation function to produce an output [18]. The typical form of single neurons is shown in Figure (1) and mathematical model showing as follows.

![Figure (1): Artificial Neuron.](image)

Neurons are arranged in layers; the ones of the same layer usually operate in a similar way and have a similar activation function. The neurons in the any layer can be associated with the neurons in other layers. Network architecture is a concept given to the arrangement of neurons into layers and the nature of the interaction inside and between layers [19]. The layers in the network can be divided basically into three types: input, hidden and output layers, as showing in Figure (2).
In 1986, Rumelhart, Hinton and Williams proposed the Back Propagation (BP) algorithm, the most popular supervised learning technique in the ANN for training MLP [20]. BP Learning consists primarily of two passes through the different layers of the network, a forward pass and a backward pass. The network's synaptic weights are all set during the forward passes. During the backward pass, the synaptic weights are all modified in compliance with the error-correction algorithm. Throughout the training cycle, the learning rate and momentum are modified to get the network out of its local minima and to promote network convergence. A small learning rate leads to a slow convergence of algorithms, whereas a too high rate can lead to failure [21]. The BP algorithm starts by determining the output (y) of each neuron in the network and then evaluates the Mean Square Error (E). The gradient (g) is then determined to change the weights as shown below [22]:

$$E = \frac{1}{2} \sum_k (d - y)^2$$ (1)

$$g = \frac{\partial E}{\partial w}$$ (2)

Where: k: No. of output neuron, d: Output target vector, W: Weight of network.

Table (1) shows the famous Back-Propagation algorithm training methods with their mathematical model [22].

| The type of algorithm                      | Mathematical model                              |
|--------------------------------------------|-------------------------------------------------|
| Gradient Descent (GD)                      | \( w_{t+1} = w_t - \alpha g_t \)               |
| Gradient Descent with Momentum (GDM)       | \( w_{t+1} = w_k - \alpha g_t + \mu w_{t-1} \) |
| Gradient Descent with Momentum and Adaptive Learning Rate (GDx) | \( w_{t+1} = w_t - \alpha_{t+1} g_t + \mu w_{t-1} \)
| Conjugate Gradient Fletcher–Reeves (CGF)  | \( p_0 = -g_0 \)
|                                            | \( p_t = -g_t + \beta_t g_{t-1} \)            |
|                                            | \( w_{t+1} = w_t + \alpha_t p_t \)            |
|                                            | \( \beta_t = \frac{g_t^T g_{t-1}}{g_{t-1}^T g_{t-1}} \) |
| Conjugate Gradient with Polak–Ribiere (CGP)| \( p_0 = -g_0 \)
|                                            | \( p_t = -g_t + \beta_t g_{t-1} \)            |
|                                            | \( w_{t+1} = w_t + \alpha_t p_t \)            |
|                                            | \( \beta_t = \frac{\Delta g_{t-1}^T g_t}{g_{t-1}^T g_{t-1}} \) |
| Levenberg–Marquardt (LM)                   | \( w_{t+1} = w_t - [J^T J + \mu I]^{-1} J^T e \) |
\( a \) : learning rate, \( w_{t+1} \): The new weight vector, \( \Delta I_i(t) \): Individual update value.

\( \mu \) : The momentum constant, \( \gamma \) : Constant used to change Weights of the neurons also would be updated using the PSO algorithm. The PSO method simulates the collective actions of particles traveling through a multidimensional space of quest when each particle has its location and velocity. Each particle shall be regarded as a possible solution to the optimization problem [23]. The location and velocity are typically expressed as vectors. For each time phase, the particle contrasts its current location with the objective (global / personal, best) location, changing its velocity appropriately to the target with the aid of a specific memory of the best position ever located both globally and independently [24]. The most common explanation of how the particle changes its velocity and location is shown in equation (3) and (4) respectively [25].

\[
v_{i(t+1)}(d) = wv_{it}(d) + c1r1(p_{pi}(d) - x_{it}(d)) + c2r2(g_b(d) - x_{it}(d)) \tag{3}
\]

\[
x_{i(t+1)}(d) = x_{it}(d) + v_{i(t+1)}(d) \tag{4}
\]

Where:

\( w \) : Represents the momentum weight of the dynamic "flying".

\( c1 \) and \( c2 \) are Recognized as cognitive and social variables for the algorithm.

\( r1 \) and \( r2 \) are casual numbers within [0,1] interval.

\( p_{bi} \) : the personal best position recorded by particle \( i \).

\( g_b \) : the global best position got by any particle in the population.

d : dimension (all the numbers of biases and weights) assembles the size of search area in (X-Y) dimensions which can be calculated from the following relation:

\[
Dimension = (input_{NN} \times hidden) + (hidden \times output_{NN}) + hidden_{bias} + output_{bias} \tag{5}
\]

### 3. FIELD PROGRAMMABLE GATE ARRAYS

FPGA is a logic device which contains programmable switches and 2D-arrays of common logic cells. FPGA has the ability to perform many tasks where it is used in complex operations, some applications, medical devices, automated systems and mathematical operations such as multiplication, division, subtraction, addition and others [26]. The Input Output Blocks (IOBs), programmable interconnects and Configurable Logic Block (CLB) is the basic components of FPGA [27].

CLB Consists of a variety of tiny logic modules that are simple construction units which have interfaces inside the CLB via a local programmable interconnect. In addition to memory and other tools, large FPGAs have tens of thousands of Configurable Logic Blocks. It is defined by a more robust overall architecture that can be implemented and can also be reprogrammed [28]. Connecting logic blocks to fundamental CLBs inside the global programmable interconnection row / column [28].

The logic module consists of the Lock Up Table (LUT) and the corresponding logic. LUT is a type of memory that can be reprogrammed and used to construct a flip-flop and a Sum of Product (SOP) combination logic function [27].

To program or configure the FPGA board there are two ways: schematic editor or by Hardware Description Language (HDL) such as Verilog language and VHDL [29]. Xilinx, Altera, Atmel and Lattice are the types of FPGAs according to their manufacturers, but Altera and Xilinx FPGA are more common types. Many versions different in their features and specifications are provided by Xilinx FPGA, Spartan FPGAs are one of them for high volume and low-cost applications, Xilinx Spartan kits are the typical choice. It refers to logic elements as slices and comprises, in general, 4-input LUTs, flip-flops, multiplexers and the other circuitry [30][31].

This paper presents the Co-Simulation of the of the trained Artificial Neural Network after generating the HDL Code on Xilinx Spartan 6 FPGA XC6SLX9FTG256 using Xilinx System Generator (XSG) with MATLAB R2012b Simulink and the Hardware Implementation on the Xilinx Spartan 6 FPGA XC6SLX9FTG256 using ISE Design Suite 14.7 program. FPGA card which will be used is shown in Figure (3).
4. THE PROPOSED CONTROLLER DESIGN

The proposed controller in this work has four inputs; of solar system which provides DC output voltage, wind system which provides the same output voltage of the solar system, A summing circuit with voltage regulator system to provide the output DC voltage when the wind speed or solar irradiation of the sun is not high enough to provide the rated voltage to the consumer and charging the battery subsystem in the meantime, The intelligent controller (Artificial Neural Network implemented on FPGA) with MPPT algorithm is designed to determine the best system to supply the consumer. When the former systems don’t provide sufficient output to supply the consumer the battery system provide the desired output until any of the systems starts to deliver the required output. This model is designed with input set at a threshold voltage equals to 70 volts, so when any input >70 its input to ANN is (1), while when the input <70 the input to ANN is (0). The artificial neural network training data set are illustrated in table (2).

Table (2): The input-output training data for ANN.

| Input | Output |
|-------|--------|
| Solar | Wind   | Sum    | Battery | Solar-S | W-S  | SUM-S | Battery-S |
| 0     | 0      | 0      | 0       | 0       | 0    | 0     | 0         |
| 0     | 0      | 0      | 1       | 0       | 0    | 0     | 1         |
| 0     | 0      | 1      | 0       | 0       | 0    | 1     | 0         |
| 0     | 0      | 1      | 1       | 0       | 0    | 1     | 0         |
| 0     | 1      | 0      | 0       | 0       | 1    | 0     | 0         |
| 0     | 1      | 0      | 1       | 0       | 1    | 0     | 0         |
| 0     | 1      | 1      | 0       | 0       | 1    | 0     | 0         |
| 0     | 1      | 1      | 1       | 0       | 1    | 0     | 0         |
| 1     | 0      | 0      | 0       | 1       | 0    | 0     | 0         |
| 1     | 0      | 0      | 1       | 1       | 0    | 0     | 0         |
| 1     | 0      | 1      | 0       | 1       | 0    | 0     | 0         |
| 1     | 0      | 1      | 1       | 1       | 0    | 0     | 0         |
| 1     | 1      | 0      | 0       | 1       | 0    | 0     | 0         |
| 1     | 1      | 0      | 1       | 1       | 0    | 0     | 0         |
| 1     | 1      | 1      | 0       | 1       | 0    | 0     | 0         |
| 1     | 1      | 1      | 1       | 1       | 0    | 0     | 0         |
The output is the enable of the switches networks that deliver the output to the consumer. The artificial neural network has been trained with two algorithms to determine the fastest algorithm with the best mean square error and minimum neurons number in the hidden layer, so using MATLAB m-file training program have been written using Levenberg-Marquardt (LM) algorithm for the data set in table (2) and the results calculated. Also, the Particle Swarm Optimization (PSO) training algorithm code has been written in MATLAB m-file for the same numbers of hidden neurons tested in the LM algorithm and the results calculated. The training data set chosen According to the weather in Iraq and the long hours of solar irradiation as we proposed previously the Solar System will be the main source then the Wind System, the Sum and finally the Battery System, the Artificial Neural Network will decide the suitable output. Figure (4) shows flowchart of the proposed controller.

![Flowchart of The Proposed Controller](image)

Figure (4): Flowchart of The Proposed Controller.

The solar system designed using MATLAB/Simulink program where the solar panel block parameter set to 36 series modules connected per string each module contains 60 diodes and a DC to DC boost converter to keep the output of the solar system as smooth as possible, Figure (5) shows the solar system.
The wind system consists of several stages to generates DC output; the first stage is the wind turbine which speed has been set to 9m/s, Permanent Magnet Synchronous Generator (PMSG) to produce three phase AC power and its parameters set to frequency of 50Hz, 6 poles magnet, 30 turns and field of 0.0207862 Wb/m², the third stage is thyristor bridge rectifier where the AC power converted to DC and finally the last stage is DC to DC converter to maintain the output smooth. Figure (6) shows the block diagram of the wind system.

5. RESULTS OF TRAINING ANN

The results of training the ANN with LM and PSO algorithms shows that the best training algorithm for the Artificial Neural Network is the Levenberg-Marquardt with single hidden layer that consist of six neurons, as seen in table (3) and Figure (7).

| No. Of Hidden Layer Neurons | Training Algorithm | Performance MSE | Epoch |
|-----------------------------|--------------------|-----------------|-------|
| 4                           | LM                 | 0.00052176      | 5     |
|                             | PSO                | 0.015625        | 40    |
| 6                           | LM                 | 1.482e-07       | 1     |
|                             | PSO                | 0.015625        | 58    |
| 9                           | LM                 | 0.0021661       | 5     |
|                             | PSO                | 0.015625        | 85    |
we used the genism(net) code to covert the Artificial Neural Network to Simulink Block the we build the switching system that will be controlled using the network, then the Simulink model of the trained ANN have been connected to the system as shown in Figure (8).

Figure (7): Best Training Results.

Figure (8): The Hybrid System with the proposed controller.
The ANN control the Switches ON and OFF according to the MPPT algorithm at a threshold voltage, notice that the inputs to the controller are 16-bit numbers and the output is one bit [0 or 1] the system has been tested and the results were remarkable. After that the Controller block have been converted to vhdl code to be later downloaded on the FPGA card.

6. CO-SIMULATION RESULTS OF THE CONTROLLER

By connecting the ISE14.7 with the MATLAB 2012b and using Xilinx System Generator a new Co-Simulation project have been created to test the controller Figure (9) shows the Co-Simulation of the controller and Figure (10) shows the scope that has been connected on the output of the Controller to calculate the time response of the controller, the results have showed that the time response between transections of the switches is 5µs.

Figure (9): Co-Simulation Results of the Controller.

Figure (10): The Controller Time Response.
7. IMPLEMENTATION RESULTS
After testing the Intelligent controller, A new project has been started opened in the ISE Design suite 14.7 program with the name NN_Controller and the device parameter has been chosen according to the FPGA card used, then the vhdl file generated earlier would be added to the project to generate the programming file that will be downloaded later on the FPGA card. The programming process include five main steps as follows:

- Design Summary Reports.
- Synthesize XST.
- Implementation Design.
- Generate Programming File.
- Configure Target Device.

When the programming file finished the program have been downloaded successfully using the Impact program in the ISE14.7 Suit as presented in Figure (11), while table (4) presented the device utilization summary.

Figure (11): Downloading the program on the FPGA card.
Table (4): Summary of the Device Utilization.

| Logic Utilization                  | Used  | Available | Utilization | Note(s) |
|-----------------------------------|-------|-----------|-------------|---------|
| Total Number Slice Registers      | 181   | 11,440    | 1%          |         |
| Number used as AND/OR logics      | 181   |           |             |         |
| Number of Slice LUTs              | 2,531 | 5,720     | 44%         |         |
| Number used as logic              | 2,421 | 5,720     | 42%         |         |
| Number of occupied Slices         | 785   | 1,430     | 54%         |         |
| Number of LUT Flip Flop pairs used| 2,531 |           |             |         |
| Number of MUXCYs used             | 2,452 | 2,860     | 85%         |         |
| Number of bonded IOBs             | 8     | 186       | 4%          |         |

Table (5) presents Compression with other Surveys for Intelligent controller using FPGA card for Wind Solar Hybrid System.

Table (5): Comparison with other Surveys results.

| Reference No. | ANN Structure | Training Algorithm | Time response |
|---------------|---------------|--------------------|---------------|
| Proposed      | 6 neurons on the Hidden layer, Satlin Activation function | ANN-LM | 5µs |
| [21]          | 20 neurons on the Hidden layer, Sigmiod Activation function | ANN-BP | 0.4s |
| [11]          | 10 neurons on the Hidden layer, Sigmiod Activation function | NARX | 0.3s |
| [16]          | 17 neurons and 9 neurons on the first and second Hidden layers, Sigmiod Activation function | ANN-GA | 0.45s |

I. Discussion

As the distributed generation grows worldwide due to a decrease in the cost of renewable energy, many of these generation sources may try grid convergence and function as insular microgrids. In these cases, power control would be crucial for any of these microgrids to allow efficient usage of energy resources in order to gain good value by controlled usage.

In this work, An FPGA intelligent controller based on wind solar system is proposed, and simulated by MATLAB/Simulink program. Then, the controlling Artificial Neural Network was implemented on the FPGA card using ISE Design Suite 14.7 program. The Intelligent controller provide fast response to the change in weather (5µs) where it responded to the change of solar irradiation and wind speed variations within 5 microseconds. Artificial Neural Networks trained with Particle Swarm Optimization (PSO) are less efficient than the ones that trained with Levenberg-Marquardt algorithm. So, The chosen structure of the Artificial Neural Network is single hidden layer with six neurons where the mean square error is (1.482e-7) with one epoch and Field Programmable Gate Array (FPGA) are the best microprocessor for implementing the ANN with the for its high-speed, better performance and easy to design with.
References

[1] P. A. Owusu and S. Asumadu-Sarkodie, “A review of renewable energy sources, sustainability issues and climate change mitigation,” Cogent Engineering, vol. 3, no. 1. Cogent OA, 2016.

[2] H. H. Al-Kayiem and S. T. Mohammad, “Potential of renewable energy resources with an emphasis on solar power in Iraq: An outlook,” Resources, vol. 8, no. 1, pp. 1–20, Mar. 2019.

[3] U. Akram, M. Khalid, and S. Shafiq, “An innovative hybrid wind-solar and battery-supercapacitor microgrid system—development and optimization,” IEEE Access, vol. 5, pp. 25897–25912, Oct. 2017.

[4] Areeg Fadhil Hussein, “Wind-Solar Intelligent Controller System based on FPGA: Review Study,” Int. J. Eng. Res., vol. V8, no. 12, 2019.

[5] C. R. Algarín, D. S. Hernández, and D. R. Leal, “A low-cost maximum power point tracking system based on neural network inverse model controller,” Electron., vol. 7, no. 1, 2018.

[6] M. Cilimkovic, “Neural Networks and Back Propagation Algorithm,” Inst. Technol. Blanchardst., 2010.

[7] A. Wierzbicki and Y. Nakamori, Swarm Intelligent Systems. Springer, 2005.

[8] M. Ciristei and A. Parera-Ruiz, An FPGA controller for a combined solar / wind power system. IEEE, 2010.

[9] T. S. B. Damodhar and D. A. S. Kumar, “Implementation of FPGA Based Hybrid Power Generator for PV and Wind Grid Applications,” Circuits Syst., vol. 07, no. 13, pp. 4280–4290, 2016.

[10] A. Jemaa, O. Zarrad, M. A. Hajjaji, and M. N. Mansouri, “Hardware Implementation of a Fuzzy Logic Controller for a Hybrid Wind-Solar System in an Isolated Site,” Int. J. Photoenergy, vol. 2018, 2018.

[11] O. Zarrad, M. A. Hajjaji, and M. N. Mansouri, “Hardware Implementation of Hybrid Wind-Solar Energy System for Pumping Water Based on Artificial Neural Network Controller,” Stud. Informatics Control, vol. 28, no. 1, pp. 35–44, 2019.

[12] P. Anand and K. Mohana Sundaram, “FPGA Based Substantial Power Evolution Controlling Strategy For Solar and Wind Forecasting Grid Connected System,” Microprocess. Microsyst., vol. 74, Apr. 2020.

[13] M. Negnevitsky, Artificial Intelligence A Guid to Intelligent Systems, 2nd ed., vol. 123. England: Pearson Education Ltd, 2005.
Informatics Control, vol. 28, no. 1, pp. 35–44, 2019.

[22] A. H. Mohammed and H. A. R. Akkar, “Design and Implementation of Artificial Neural Network for Mobile Robot based on FPGA,” University of Technology, 2015.

[23] B. Vasumathi and S. Moorthi, “Implementation of hybrid ANNPSO algorithm on FPGA for harmonic estimation,” Eng. Appl. Artif. Intell., vol. 25, no. 3, pp. 476–483, 2012.

[24] M. Sheikhan and N. Mohammadi, “Time series prediction using PSO-optimized neural network and hybrid feature selection algorithm for IEEE load data,” Neural Comput. Appl., vol. 23, no. 3–4, pp. 1185–1194, 2013.

[25] R. G. Reynolds and B. Peng, “A tribute to C. V. Ramamoorthy: Knowledge integration on-the-fly in swarm intelligent systems,” Proc. - Int. Conf. Tools with Artif. Intell. ICTAI, vol. 1995, pp. 197–208, 2006.

[26] S. Roy, “Fundamentals of FPGA Architecture,” no. November, Technical and Scientific Publisher, 2017.

[27] T. L. Floyd, Digital Fundamentals, 9th ed. USA: Pearson Education Ltd, 2006.

[28] U. Farooq, Z. Marrakchi, H. Mehrez, U. Farooq, Z. Marrakchi, and H. Mehrez, “FPGA Architectures: An Overview,” in Tree-based Heterogeneous FPGA Architectures, Springer New York, 2012, pp. 7–48.

[29] B. Mealy and F. Tappero, Free Range VHDL, 1st ed. USA: freerangefactory.org, 2013.

[30] A. Pang and P. Membrey, Beginning FPGA: Programming Metal Your Brain on Hardware. USA: Springer New York, 2017.

[31] Xilinx, “Spartan-6 FPGA Configuration User Guide (UG380),” 2015.