FPGA-Based Implementation of Intelligent Predictor using ANN for Global Solar Irradiation

Shalini.M.G, Punithkumar.M.B, M.B.Anandaraju

Abstract— Global solar irradiation is considered as one of the most important parameter in the design of renewable and solar energy systems. Global solar irradiation is usually represented as time series. Frequently, the measured data are not always available, especially in the remote areas because of the absence of the meteorological stations or measuring instruments. Numerous studies in the literature have shown the possibility to find a correlation between solar irradiation and other meteorological parameters such as air temperature, humidity, sunshine duration etc. The models that are existing so far are based on the probability estimation, which do not always give good generation of data.in order to overcome this problem AI techniques have been applied. In this paper suitable intelligent predictor is developed and implemented using a Belgaum region database as the input using ANN for solar irradiation., to develop a hardware board which can be used for real time predictors of solar irradiation in areas where there no stations.

Keywords- Global solar irradiation, ANN

I. INTRODUCTION

In future the world will face to energy crises due to lack of fossil fuels i.e. coal, nuclear, oil, gases etc. This is a matter of concern for the entire countries whose economy heavily depend on its use of energy. Non-conventional energy resources like solar energy, wind energy, tidal and biomass are the second option to generate energy and its utilization. Regarding non conventional energy sources sun plays a important role for human beings so sun's heat and light provide an abundant source of energy that can be harnessed in many ways. There are a different variety of technologies that have been developed to take advantage of solar energy. The prediction of hourly solar radiation data has important consequences in many solar applications. Such data can be regarded as a time series and its prediction depends on accurate modeling of the stochastic process. The computation of the conditional expectation, which is in general non-linear, requires the knowledge of the high order distribution of the samples. Using a finite data, such distributions can only be estimated or fit into a pre-set stochastic model. Methods like Auto−Regressive (AR) prediction, Markov chains and ARMA model for designing the non-linear signal predictors are examples to this approach. The neural network (NN) approach also provides a good to the problem by utilizing the adaptive nature. Since NN’s can be trained to predict results from examples, they are able to deal with non−linear problems.

Global solar irradiation is usually presented as a time series and is especially challenging to predict its values in situations where an underlying model for generating data is not known. Frequently, these data (measured data) are not always available, especially in remote areas because of the absence of meteorological stations or measuring instruments. For this purpose, several models have been developed in order to generate these data based on stochastic models such as AR (auto-regressive), ARMA (AR-moving average), ARIMA (AR integrated MA) and Markov chains. However, these models are based on the probability estimation, which do not always give good generation of the data. Therefore, in order to overcome this problem, Artificial Intelligence (AI) techniques (neural networks, fuzzy logic and hybrid neural networks) have been applied with success for the modeling, prediction and forecasting of solar radiation data . The advantage of these techniques over standard mathematical models is that they do not require the knowledge of internal system parameters, involve less computational effort, and offer a compact solution for multi-variable problems.in this paper intelligent predictor has been developed using a database of Belgaum region as a input to implement on FPGA to get a hardware board used in the real time applications. The main objective if this paper is to develop and implement a intelligent MLP-predictor using ANN for solar irradiation into a configurable FPGA hardware.

II. FUNDAMENTALS FOR IMPLEMENTATION

A. *Elementary neuron*

The neuron function consists in calculating the pondered sum function ($\sum w_{ij}x_i$) and updating the state of the neuron by applying the activation function. The neuron has set of nodes that connect it to inputs, output, or other neurons, these nodes are also called synapses.

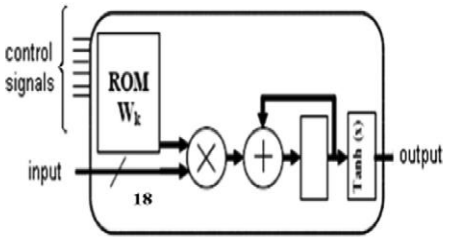

Figure 1 Neural processor architecture (the elementary neuron).

Manuscript published on 30 July 2013.
***Correspondence Author(s)**
Shalini.M.G, PG Student, VLSI Design and Embedded System, B.G.S. Institute of Technology, B.G.Nagar, Mandya-571448, Karnataka, India.
Punithkumar.M.B, Assistant Professor, Department of Electronics and Communication Engineering, B.G.S. Institute of Technology, B.G.Nagar, Mandya-571448, Karnataka, India.
Prof. M.B.Anandaraju, Professor and HOD, Department of Electronics and Communication Engineering, B.G.S. Institute of Technology, B.G.Nagar, Mandya-571448, Karnataka, India.

B. Multiply-accumulate module

Multiplication operation is an essential operation for microprocessor, digital signal processor and graphics engine. Multiplication algorithm will be used to illustrate methods of designing different cells so that they fit into larger structure. Several types of adders can be used to achieve the states of the pondered sum: combinative, in series, dynamic. In the same way, there are several ways to realize the multiplication; the most classical are: in series, series/parallel and completely parallel. However, the use of the FPGA technology imposes constraints on surface (size). In order to decrease the area (or surface) needed by a numeric multiplier, various methods are available.

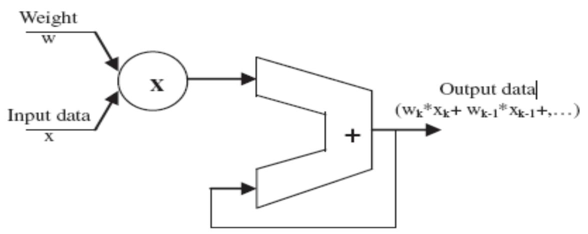

Figure 2 Multiply-accumulate module (MAC)

C. Activation function

The activation function used is Tangent-sigmoid which is very difficult and direct implementation for nonlinear sigmoid transfer functions is very expensive. There are two practical approaches to approximate the sigmoid function with simple FPGA designs. The first one is linear approximation described by a combination of lines in the form y = ax + b which is used to approximate the sigmoid function. Note that if the coefficients for the lines are chosen to be of the power of two, the sigmoid functions can be realized by a series of shift and add operations. Many implementations of neuron transfer functions use such linear approximations. The second method is look-up tables. In this paper the first approach is chosen. So in order to simplify the function expression, this was linearized on several intervals [Ci,Ci+1] and its values are evaluated using two constants (a_i and b_i) corresponding to this intervals.

$$\begin{cases} f(x_i) = a_i x_i + b_i & \text{if } x_i \in [c_i, c_{i+1}]. \\ f(x) = 1 & \text{if } x > 3. \end{cases}$$

D. Sigmoid function

The sigmoid function is the most common form of activation function used in the construction of artificial neural networks. Whereas a threshold function assumes the value of 0 or 1, a sigmoid function assumes a continuous range of values form 0 and 1. Note also that the sigmoid function is differentiable, which is an important feature of neural network theory. The derivative of the sigmoid function has a nice property, which makes is easy to calculate:

$$\frac{\partial\varphi(v)}{\partial v} = \varphi(v)(1-\varphi(v))$$

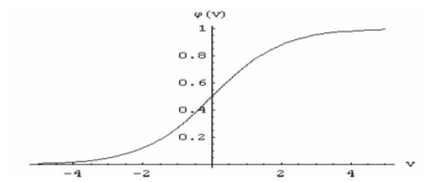

Figure 3 Sigmoid (logistic) activation function

E. Rom

Selecting weights precision is one of the important choices when implementing ANNs on FPGAs. Weight precision is used to trade-off the capabilities of the realized ANNs against the implementation cost. A higher weight precision means fewer quantization errors in the final implementations, while a lower precision leads to simpler designs, greater speed and reductions in area requirements and power consumption. In the function neuron, the weights are saved in a ROM memory; the size of this memory will be defined according to the position of the neuron vin the network (i layer), and the number of inputs. Therefore, the ROM of the input neuron store only two weights (w_1,w_2) and the bias b.
where i is the number of layers and i = 1 is the input layer.
For i > 1:

$$T_{ROM(i)}=N_{neuron(i-1)}+1$$

where: $T_{ROM(i)}$ the size of the ROM of the layer i. $N_{neurone(i-1)}$ is the number of neurons in the layer i -1.

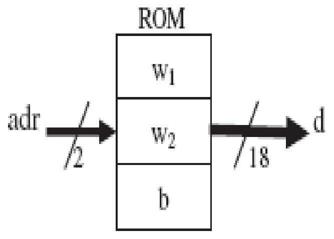

Figure 4 ROM module for the neuron in the input layer

F. Multi-layer Perceptron

The perceptron is the simplest form of a neural network used for the classification of a special type of patterns, which are linearly separable. The MLP consists of various layers: an input and output ones and between of them lie one or several hidden ones whose outputs are not observable. These layers are based on some processing unit (neurons) interconnected by means of feed-forward pondered links. All these processing units carry out the same Operation. The inputs are fed into the input layer and get multiplied by interconnection weights as they are passed from the input layer to the hidden layer.

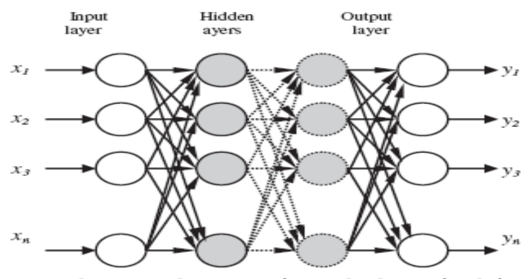

Figure 5 Schematic diagram of a multi-layer feed-forward neural network.

Within the hidden layer, they get summed then processed by a nonlinear function (usually the hyperbolic tangent). If more than a single hidden layer exists then, as the processed data leaves the first hidden layer, again it gets multiplied by interconnection weights, then summed and processed by the second hidden layer and so on. Finally the data is multiplied by interconnection weights then processed one last time within the output layer to produce the neural network output.

G. *Information processor*

The network usually consists of an input layer, some hidden layers and an output layer. In its simple form, each single neuron is connected to other neurons of a previous layer through adaptable synaptic weights. Knowledge is usually stored as a set of connection weights (presumably corresponding to synapse efficacy in biological neural systems). Training is the process of modifying the connection weights in some orderly fashion using a suitable learning method. The network uses a learning mode, in which an input is presented to the network along with the desired output and the weights are adjusted so that the network attempts to produce the desired output.

The weights after training contain meaningful information whereas before training they are random and have no meaning. The node receives weighted activation from other nodes through its incoming connections. First, these are added up (summation). The result is then passed through an activation function; the outcome is the activation of the node.

A multilayer network can mathematically approximate any continuous multivariate function to any degree of accuracy, provided that a sufficient number of hidden neurons are available. Thus, instead of learning and generalizing the basic structure of the data, the network may learn irrelevant details of individual cases.

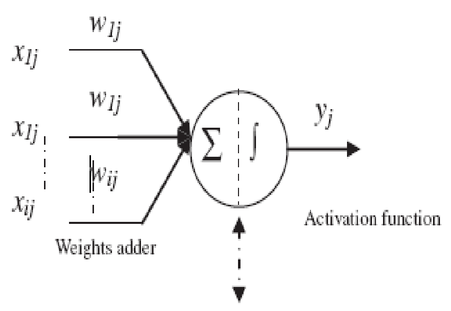

Figure 6 Information processing in a neural network unit.

$$y_j = f\left[\left(\sum_i w_{ij} x_{ij}\right) + b_j\right]$$
$$f(y_i) = \frac{2}{(1+e^{-2y_i})} - 1$$

Where y_j is the output of the processing unit, w_{ij} the synaptic weight coefficient of the ith input of the processing unit, b_j is the bias.

The sigmoid function generates a continuous output between 0 and 1 as the neuron's net input goes from negative to positive infinity. It is defined as a strictly increasing function that exhibits smoothness and asymptotic properties. The sigmoid function is differentiable, which is an important feature of neural network theory.

H. *Parallelism implementation*

In order to develop the neuron in its numeric format, the three main elements i.e. Rom, Activation function and MAC is used. Based on the level of parallelisms given by various configurations, the neural network can be classified into two groups:

- The first one is called configuration of down-level parallelism, which consists of implementing completely the operation of the network based only on one neuron, and a unit of control; this last reconfigure the neuron by the modification of the weights and the inputs of the network. The advantage of this technique is that it is less gluttonous in surface; the inconvenience is the risk of performance loss (speed of calculation).
- The second is called configuration of high-level parallelism, it is based on the interconnection of all neurons of the network architecture, this type of development is solely valid for networks of small size (according to the size of the targets circuit).The advantage of this technique is the quick calculation performance (computing time). This is the configuration used in this project.

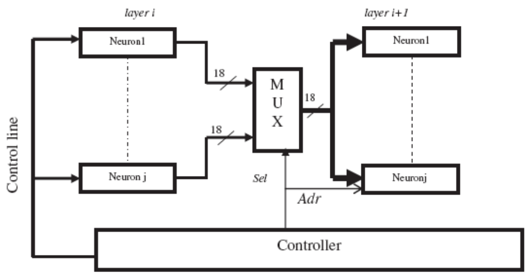

Figure 7 Parallel implementation of a neural network

The hardware implementation of the MLP on FPGA is based on two main elements . The first one is the controller, and the second is the neuron function. The controller is like a state machine, which links the necessary calculation steps in order to reproduce a propagation of the MLP network. It consists of a management module of the control signals (CE0, CE1 and Load), and a module generation of the signals for data selection (SEL0, and SEL1) through the multiplexers, and the addresses signals (ADR0, and ADR1) of the ROM containing the weights and bias.

I. *Back propagation algorithm*

The back propagation algorithm is used in layered feed-forward ANNs. This means that the artificial neurons are organized in layers, and send their signals "forward", and then the errors are propagated backwards. The network receives inputs by neurons in the input layer, and the output of the network is given by the neurons on an output layer. There may be one or more intermediate hidden layers.

J. *Learning process*

Among the many interesting properties of a neural network is the ability of the network to learn from its environment and to improve its performance through learning. A neural network learns about its environment through an iterative process of adjustments applied to its synaptic weights and thresholds. Learning is a process by which the free parameters of a neural network are adapted through a continuing process of stimulation by the environment in which the network is embedded. The type of learning is determined by the manner in which the parameter changes take place. The network uses a learning mode, in which an input is presented to the network along with the desired output and the weights are adjusted so that the network attempts to produce the desired output. The weights after training contain meaningful information whereas before training they are random and have no meaning. Training is the process of modifying the connection weights in some orderly fashion using a suitable learning method.

III. IMPLEMENTATION

The different steps for implementing an intelligent predictor for solar irradiation from other meteorological parameters are,

1. Define the inputs and the output of the employed MLP for different proposed configurations: [G=~f(t, T,S,RH);G=~f(t,T,S);G=~f(t,T,RH);G=~f(t,S,RH);G=~f(t,T)and G = ~f (t,S)]. Where t is the time (day of year).
2. Select the best configurations according the simulation results carried out in Matlab, based on the accuracy of the results compared with measured data as well as the economic confiscation (i.e., the configuration which does not need several sensors in its inputs for predicting and measuring irradiation, and simple MLP architecture).
3. Design the developed configuration through hardware description languages (VHDLs), which is chosen in this study.
4. Test and simulate the developed configuration using ModelSim.
5. Design implementation.

A. *MLP-based prediction of global solar irradiation*

A MLP is employed for prediction of global solar irradiation from other meteorological data. The different MLP-configurations used are:

1. G = ~f (t,T,S,RH) has as input parameters the air temperature, sunshine duration, relative humidity and the day of year, while the output is the global solar irradiation, shown in Figure 8(a).
2. G=~f (t,TS) has as input parameters the air temperature, sunshine duration, and day of year, while the global solar irradiation in the output, shown in Figure 8(b).
3. G=~f (t,T,RH) has as input parameters the air temperature, relative humidity, and day of year, while the output is the global solar irradiation, shown in Figure 8(c).
4. G=~f (t,RH,S) has as input parameters the relative humidity, sunshine duration and day of year, while the output is the global solar irradiation, shown in Figure 8(d).
5. G=~f (t,T) has as input parameters the air temperature and day of year, while the output is the global solar irradiation, shown in Figure 8(e).
6. G=~f (t,S) has as input parameters only the sunshine duration and the day of year while the output is the global solar irradiation, shown in Figure 8(f).

Therefore, the inputs and output are fixed initially for each configuration; however, the number of hidden layers and the neurons within these layers are optimized during the learning process based on the value of the Root Mean Square Error (RMSE).

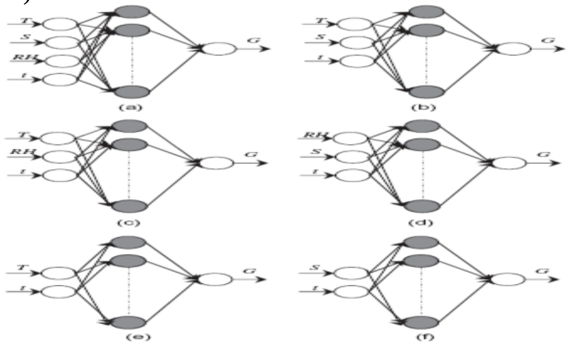

Figure 8 The different MLP-predictors: (a) G = ~f (t,S,T,RH), (b) G =~f (t,T,S), (c) G =~f (t,T,RH), (d) G =~f (t,S,RH), (e) G =~f (T,t) and (f) G =~f (S,t).

It should be noted from Table 1that one hidden layer within four neurons is sufficient for predicting the global solar irradiation data. The simplest architecture can be implemented very easily and does not need a lot of space on the FPGA.

Table 1:Statistical test between measured and predicted global solar irradiation by MLP for different configurations

MLP-configuration	Architecture:input/hidden/output	RMSE	MBE	r
(1) G=f(t,T,S,RH)	4*4*1	0.00110	4.836	0.9720
(2) G=f (t,T,S)	3**4*1	0.00110	4.622	0.9744
(3) G=f(t,T,RH)	3*4*1	0.00290	6.448	0.8978
(4) G=f (t,S,RH)	3*4*1	0.00120	4.717	0.9731
(5) G =f (t,T)	3*4*1	0.00414	6.3430	0.8927
(6) G=f (t,S)	3*4*1	0.00127	4.7156	0.9724

Correlation between global solar irradiation and sunshine Duration The relations between the fraction (G/G0) and sunshine duration (S/S0) are:

$$\frac{G}{G_0} = a + b\left(\frac{S}{S_0}\right)$$

$$\frac{G}{G_0} = a + b\left(\frac{S}{S_0}\right) + c\left(\frac{S}{S_0}\right)^2$$

where a, b and c are the regression coefficients. Figure 9 shows the correlation between daily global radiation and sunshine duration. The correlation coefficient is 94%.

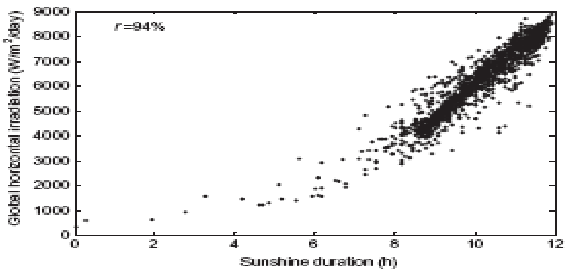

Figure 9 Correlation between daily global horizontal radiation and sunshine duration

Correlation between global solar irradiation and air temperature. Figure10 shows the correlation between the global solar irradiation and the air temperature. In order to evaluate the correlation model between the air temperature and the global solar irradiation received on a horizontal surface, we have considered the data measured from sunrise until midday and the data from midday until sunset.

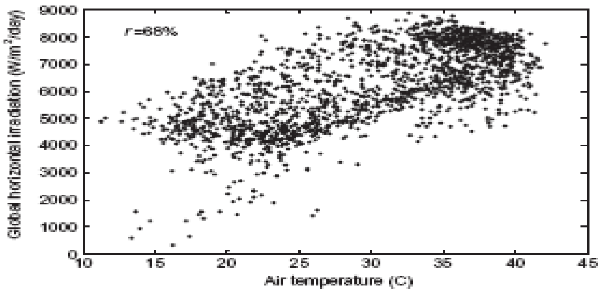

Figure 10 Correlation between daily global horizontal radiation and air temperature

The linear regression for the experimental data is given by:

$$\frac{G}{G_0} = a_1 + b_1\left(\frac{T}{T_{\max}}\right)$$

where a_1 and b_1 are the coefficients of linear regression. The correlation coefficient is 68%, which is less than that mentioned above (in the case of global solar irradiation and sunshine duration).

Correlation between global irradiation and relative humidity Figure 11 shows the correlation between the global solar irradiation and the relative humidity site.

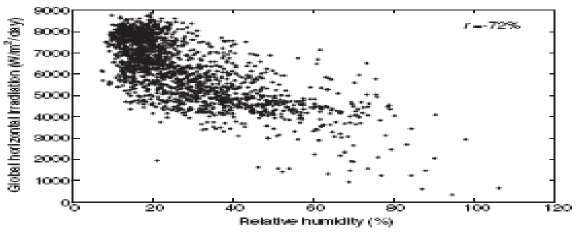

Figure 11 Correlation between daily global horizontal radiation and relative Humidity.

The correlation model is given by the following formula:

$$\frac{G}{G_0} = a_2 + b_2\left(\frac{RH}{RH_{\max}}\right)$$

where a_2 and b_2 are the coefficients of linear regression. The correlation coefficient is 72%.

IV. RESULTS AND DISCUSSION

Simulation result using Matlab Tool For several simulations, the best performance for each configuration as well as the statistical test between measured and predicted global solar irradiation. Figure shows a comparison between the measured and the predicted global solar irradiation for the different configurations. As can be seen from these curves, the obtained results by the first, second, fourth and sixth configurations are very close with the actual data, which the network has not seen before.

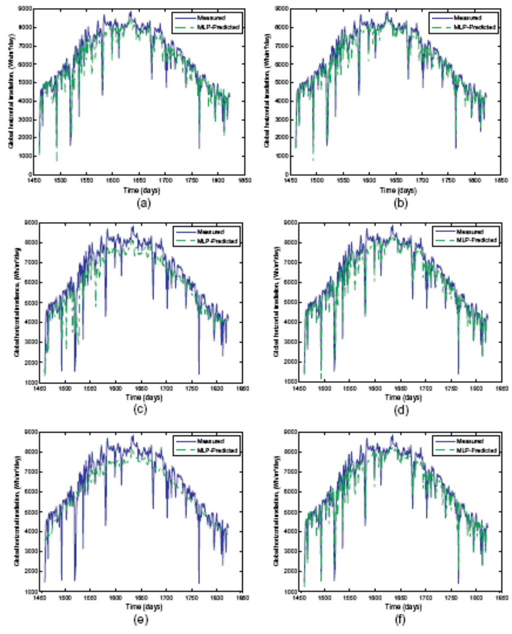

Figure 12 Comparison between measured and predicted global solar radiation data for different configurations: (a) G=f(t,T,S,RH), (b) G=f (t,T,S), (c) G=f (t,T,RH), (d) G=f (t,S,RH), (e) G=f (t,T), and (f) G=f(t,S).

The rest of the configurations (the third and fifth) cannot provide very acceptable results. It should be noted from Table that one hidden layer within four neurons is sufficient for predicting the global solar irradiation data. We can conclude that the sunshine duration increases clearly the performance of the predictor, however, the relative humidity cannot influence the accuracy.

Table2 :Analysis of the best configurations.

Configuration	Accuracy	Simplest architectue	Availability, cost and efficiency
G=f(t,T,S,RH)	0.9720	4*4*1	Expensive (sunshine duration sensor), not always available and simplest architecture.
G=f(t,T,S)	0.9749	3*4*1	Expensive (Sunshine duration sensor), not always available, accurate and simplest architecture.
G=f (t,T,RH)	0.8978	3*4*1	Not expensive, sensors always available, not accurate, and simplest architecture.
G=f(t,S,RH)	0.9730	3*4*1	Expensive (sunshine duration sensor), not always available, accurate and simplest architecture.
G=f(t,T)	0.8927	2*4*1	Expensive (sunshine duration sensor), not always available, accurate and simplest architecture.
G=f(t,S)	0.9724	2*4*1	Expensive (Sunshine duration sensor), not always available, accurate and simplest architecture.

However, the problem now is to choose the best and more suitable configuration for renewable energy applications and other fields (climate, agriculture, etc.).

Therefore, in order to select which one will be used for implementing into FPGA the following criteria are defined:

- Accuracy (correlation coefficient).
- Simplicity of the architecture (number of hidden layers, and neurons within hidden layers).

- Availability and the cost of the sensors used (which one is cheaper, T, RH or S sensors, and most available).

Based on the above criteria and according to the results presented in Table 2, we can select two configurations; the second and the third. Therefore the former [G=f(t,T,S)] can provide more accurate results but it is expensive and the sensor required for sunshine duration is not always available, especially in underdevelopment countries and the latter [G=f (T,t,RH)] is not as accurate, but can provide acceptable results, is not expensive and the sensors used (air temperature and relative humidity) are always available. In fact, the former is very suitable for applications which need more accuracy, while the latter for applications that do not need very accurate results. Therefore, the developed MLP-predictors can be given by the following formula:

$$\tilde{y}=\sum_{k=1}^{M}\left\{\left(\frac{2}{1+\exp\left(-\sum_{j=1}^{M}\left(\sum_{i=1}^{N}w_1(i,j)x(i)+b_1\right)\right)}-1\right)\right\}w_2(k)b_2(k)$$

where w_1, w_2, b_1 and b_2 are the weights and bias of the networks, respectively, x represents the inputs data which can be the sunshine duration, air temperature, and day of year or relative humidity (according to the predictor). ~y is the predicted global solar irradiation, M and N are the number of neurons in the hidden layer and in the input layer, respectively. Once both configurations [G=f(t,S,T) and [G=f (t,T,RH)] are optimized with respect to the number of hidden layers and the number of the neurons within each layer, the weights and bias are saved for each configuration in order to be used for the implementation of the MLP-predictor on the FPGA.

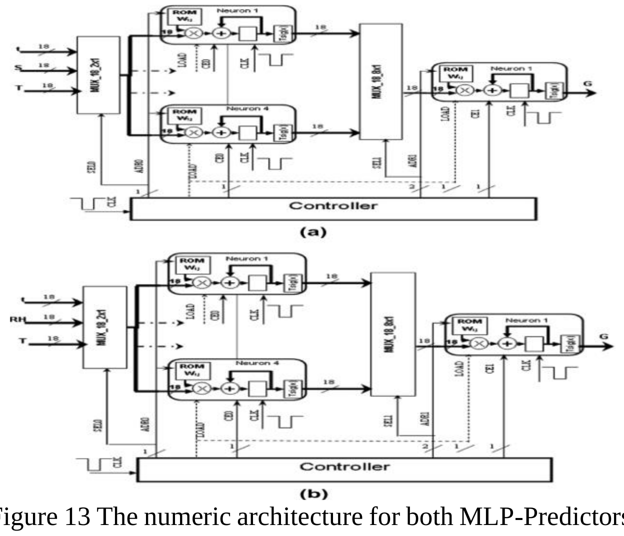

Figure 13 The numeric architecture for both MLP-Predictors: (a) G=f(t,T,S) and (b) G=f(t,T,RH).

The numeric architectures of these predictors are shown in Figure 13(a) and (b). They contain 3 + 4 + 1 elementary neurons, Mux_12x2x1, Mux_18x8x1, Mux_18x16x1 and the control unit. The simulation result by using the ModelSim.

A. VHDL-simulation results

Once both configurations G= (t,T,S), G= (t,T,RH)are optimized with respect to the number of hidden layers and the number of the neurons within each layer, the weights and bias are saved for each configuration in order to be used for the implementation of the MLP-predictor on the FPGA. The ISE Ver. 10.1 of Xilinx is used in the present study for developing the VHDL source code. Soft computing programs were written by using the VHDL and the different data used in this study have been coded on 18 bits in fixed-point (as described above). To be more specific, this can be easily represented as std_logic_vectortype, which is the most suitable data type for digital processing in an FPGA, and then the ModelSim software is used for simulation of the developed configurations on VHDL. The simulation result by using the ModelSim is presented in Figure 14,where T is the air temperature, S is the sunshine duration, RH is the relative humidity and G is the global solar radiation. A comparison between MatLab and VHDL results is presented in Figure 15(a) and (b) for the designed MLP-predictors G= f (t,T,S), G= f (t,T,RH). As can be seen a good agreement is obtained between both series and also the correlation coefficient r is about 99%. These results show the effectiveness of the implemented MLP-predictors on VHDL. Subsequently, by using the Virtex-II FPGA (XC2v1000) board of Xilinx the proposed architectures will be implemented and tested.

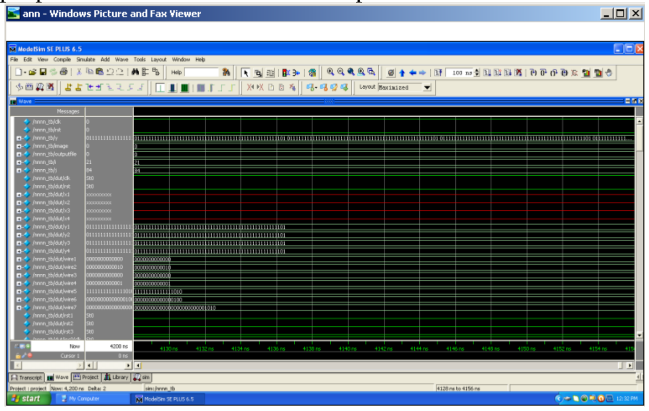

Figure 14 Simulation results by using ModelSim

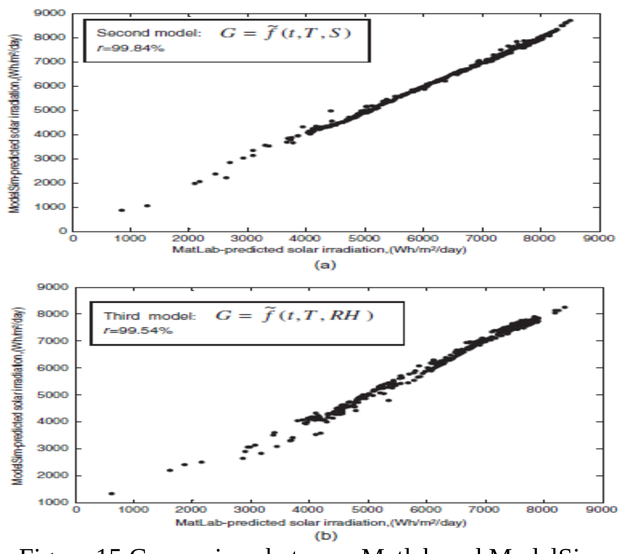

Figure 15 Comparison between Matlab and ModelSim results: (a)G= f (t,T,S), (b)G= f (t,T,RH)

B. FPGA-implementation of MLP-predictors

The implementation of the designed MLP-predictors G= f (t,T,S), G= f (t,T,RH)was achieved using a Xilinx_ Virtex-II FPGA (XC2v1000) from Xilinx. This FPGA has 3072 slices and 6114 logic cells. The FPGA has also thirtytwo18 _ 18 multipliers, as well as thirty-two 18 K bit modules of dedicated dual-port RAM. Figure 16 shows the Register Transfer Level (RTL) schematic of both MLP-predictors.

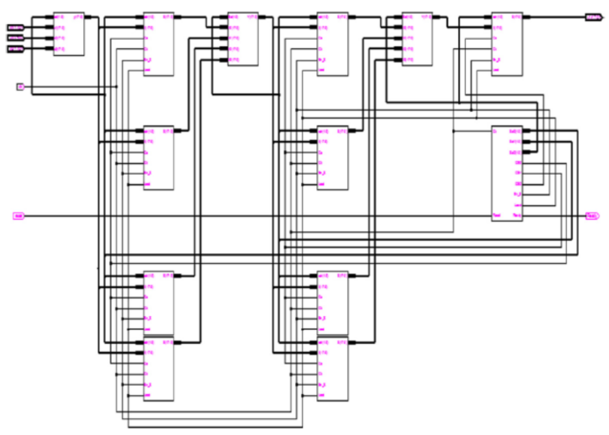
Figure 16 The Register Transfer Level (RTL) schematic of both MLP-predictors for both predictors(a)G=$\tilde{f}$(t,T,S), (b)G=$\tilde{f}$(t,T,RH)

V. CONCLUSION AND FUTURE ACTIONS

In this paper, intelligent predictors for solar irradiation are proposed and implemented into reconfigurable FPGA Virtex-II (XC2v1000) using a Belgaum region database as input. A MLP network is used for developing the solar irradiation predictor based on some meteorological data (air temperature, sunshine duration and relative humidity). The accuracy and the generalization of the MLP in the system prediction have been demonstrated by comparing the test results with actual data. Two MLP-predictors have been chosen based on cost, accuracy and availability of the sensors. The proposed predictors are suitable for the particular area considered climatic conditions; however, they can also be used in areas that have similar environmental conditions. Therefore, in order to develop a semi-global predictor, a large database would be used for several areas (varied climatic conditions). The proposed hardware devices can be considered as sensors for measuring the global solar irradiation from air temperature, relative humidity or sunshine duration in areas where there are no instruments for measuring the solar irradiation data. Some other geographical parameters (latitude, longitude, Albedo, etc.) may improve the accuracy of the proposed predictors. The designed predictors have huge potential in industrial electronics for developing intelligent sensors for solar irradiation based on the FPGA hardware.

Future work:

- Can improve the accuracy of the designed predictor by using a large dataset and introducing some other parameters.
- Modification of the predictor time at 2 hour,15 minutes etc.,

REFERENCES

1. Adnan, S., Arcakly´ ogclu, E., Ozalp, M., &Agclar, N. C. (2005). Forecasting based on neural network approach of solar potential in Turkey. *Renewable Energy*, 30, 1075–1090.
2. Aguiar, R. J., &Collares-Perreira, M. (1992). T.A.G.A time dependent autoregressive Gaussian model for generating synthetic hourly radiation. *Solar Energy*, 49(3), 167–174.
3. Allen, R. G. (1997). Self-calibrating method for estimating solar radiation from air temperature. *Journal of Hydraulic Engineering*, 2(2), 56–67.
4. Amos, B., Omondi, C., &Aajapakse, J. (2006). FPGA implementations of neural networks. *Springer*.
5. Angstrom, A. (1924). Solar and terrestrial radiation. *Quarterly journal of the Royal Meteorological Society*, 50, 121–126.
6. Benghanem, M., Mellit, A., &Alamri, S. N. (2009). ANN-based modelling and estimation of daily global solar radiation data: A case study. *Energy Conversion and Management*, 50, 1644–1655.
7. Bristow, K. L., & Campbell, G. S. (1984). On the relationship between incoming solar radiation and daily maximum and minimum temperature. *Agricultural and Forest Meteorology*, 31, 159–166.
8. Cao, J. C., & Cao, S. H. (2006). Study of forecasting solar irradiance using neural networks with preprocessing sample data by wavelet analysis. *Energy*, 3, 3435–3445.
9. Chandel, S. S., Agarwal, R. K., &Pandey, A. N. (2005). New correlation to estimate global solar radiation on horizontal surfaces using sunshine hour and temperature data for indian sites. *Journal of Solar Energy Engineering*, 127, 417–420.
10. David, B. A., &Atsu, S. S. D. (1999). Estimation of solar radiation from the number of sunshine hours. *Applied Energy*, 63, 161–167.
11. Gomez, V., & Casanovas, A. (2003). Fuzzy modeling of solar irradiance on inclined surfaces. *Solar Energy*, 75, 307–315.
12. Haykin, S. (1999). Neural network: A comprehensive foundation. Neural Network: *Macmillan.*
13. Helen, C. (2007). Power estimating clear-sky beam irradiation from sunshine duration. *Solar Energy*, 71(4), 217–224.
14. Hontoria, L., Aguilera, J., &Zufiria, P. (2005). An application of the multilayer perceptron: Solar radiation maps in Spain. *Solar Energy*, 79, 523–530.
15. Maafi, A., &Adane, A. (1989). A two state Markovian model of global irradiation suitable for photovoltaic conversion. *Solar Wind Technology*, 6, 247–252.
16. Mehreen, S., Muneer, G. T., &Kambezidis, H. D. (1998). Models for obtaining solar radiation from other meteorological data. *Solar Energy*, 64(1–3), 99–108.
17. Mekki, H., Mellit, A., Salhi, H., &Khaled, B. (2008). FPGA-based implementation of multilayer perceptron for modelling of photovoltaic panel. *American Institute of Physics A.I.P Conference Proceedings*, 1019, 211–215.