FPGA-based implementation of intelligent predictor for global solar irradiation, Part I: Theory and simulation

A. Mellit, H. Mekkia, A. Messai, S.A. Kalogirou

Keywords:
Solar irradiation
Intelligent predictor
MLP
VHDL
FPGA

ABSTRACT

Recent advances in artificial intelligent techniques embedded into a Field Programmable Gate Array (FPGA) allowed the application of such technologies in real engineering problems (robotic, image and signal processing, control, power electronics, etc.), however, the application of such technologies in the solar energy field is very limited. The embedded intelligent algorithm into FPGA can play a very important role in energy and renewable energy systems for control, monitoring, supervision, etc. In this paper, the software as well as the implementation of intelligent predictors for solar irradiation on reconfigurable FPGA is described. FPGA technology was employed due to its development, flexibility and low cost. An experimental dataset of air temperature, solar irradiation, relative humidity and sunshine duration in a specific area is used; this database has been collected from 1998 to 2002 at Al-Madinah (Saudi Arabia). Initially, a MultiLayer Perceptron (MLP) is trained by using a set of 1460 patterns and then a set of 365 patterns are used for testing and validating the MLP-predictor. Six MLP-predictors (configurations) are proposed and developed by varying the MLP inputs data, while the output is always the global solar irradiation for different configurations $G = f(t, T, RH), G = f(t, T), G = f(t, RH), G = f(t, RH)G = f(t, T)$ and $G = f(t, S)$. Subsequently, the different MLP-predictors developed are written and simulated under the Very High Speed Integrated Circuit Hardware Description Language (VHDL) and ModelSim. The best designed architecture for different MLP-predictors is then implemented under the Xilinx Virtex-II FPGA (XC2v1000). The developed hardware devices permit the prediction of global solar irradiation using available air temperature, relative humidity and sunshine duration; therefore, the designed configurations are very suitable especially in areas, where there are no instruments for measuring the solar irradiation data.

© 2010 Elsevier Ltd. All rights reserved.

1. Introduction

Global solar irradiation is considered as the most important parameter in the design of renewable and solar energy systems, particularly for the sizing of photovoltaic (PV) systems. Numerous studies in the literature have shown the possibility to find a correlation between solar irradiation and other meteorological parameters such as air temperature, sunshine duration, relative humidity and cloud cover (Angstrom, 1924; Allen, 1997; Bristow & Campbell, 1984; Chandel, Agarwal, & Pandey, 2005; Mehreen, Muneer, & Kambezidis, 1998; Prieto, Martinez-Garcia, & Garcia, 2009; Tymvios, Jacobides, Michaelides, & Scouteli, 2005).

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 chain (Aguir & Collares-Perreira, 1992; Maafi & Adane, 1989; Mora-Lopez & Sidrach-de-Cardona, 1998; Santos & Pinazo, 2003;). 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 modelling, prediction and forecasting of solar radiation data (Mellit & Kalogirou, 2008). 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 (Mellit & Kalogirou, 2008). In this context, several studies on the use of...
neural networks have been published in the literature (Adnan, Arckakyoglu, Ozalp, & Agclar, 2005; Cao & Cao, 2006; Hontoria, Aguilar, & Zufiria, 2005; Mellit, Benghanem, Hadj Arab, & Guessoum, 2005; Mellit, Benghanem, & Kalogirou, 2006; Mellit, Kalogirou, Shari, Salhi, & Hadj Arab, 2008; Sozen, Arckakyoglu, & Ozalp, 2004). Additionally, fuzzy logic has been used for estimation and prediction of solar irradiation. Sen (1998) used a fuzzy logic for estimating solar radiation from sunshine duration measurements. The fuzzy approach is applied for three sites with monthly average daily irradiances in the western part of Turkey. A fuzzy model of solar irradiance on inclined surfaces has been developed by Gomez and Casanovas (2003). The fuzzy model includes concepts from earlier models, though unlike these, it considers non-disjunctive sky categories. The proposed model offers performance similar to that of the models with the best results in the comparative analysis of the literature, such as the Perez model. David and Atsu (1999) used several expressions to estimate global solar radiation from sunshine hours for weather stations in Oman; these include linear, modified linear algorithm, Angstrom-Black type regression, a quadratic function and a new Angstrom-Black type algorithm. A linear model for estimating clear-sky beam radiation from observed radiation and sunshine duration is proposed by Helen (2007). Mellit, Benghanem, Hadj Arab A and Guessoum (2004, 2007) applied the ANN and ANFIS (Adaptive Neuro-Fuzzy Scheme) for the prediction of daily solar radiation from mean temperature and sunshine duration. An extended survey of the application of artificial intelligence techniques for modelling and forecasting solar radiation can be found in Mellit et al. (2004).

Rehman and Mohandes (2008) used an ANN for the estimation of daily solar radiation from air temperature and relative humidity at Abha City (Saudi Arabia). A set of four years are used for training the network, while a set of one year is used for testing and validating the model. The absolute mean percentage error for the case when only one day of the year and mean temperature were used as inputs was 11.8%. This error reduces to about 10.3% when maximum temperature is used instead of mean temperature. Results show that using the relative humidity along with daily mean temperature outperforms the other cases with absolute mean percentage error of 4.49%.

Recently, a comparison between different linear, polynomial regression and MLP-models for estimating the global solar radiation at Al-Madinah city was presented by Benghanem, Mellit, and Alamri (2009). The data used in this work are the global irradiation \( G \), diffuse irradiation \( H_d \), air temperature \( T \) and relative humidity \( RH \). Results show that the model using as inputs the sunshine duration and air temperature, gives the most accurate results with a correlation coefficient of 97.65%.

It can be concluded from above mentioned results that it is possible to find a suitable model for the prediction of solar radiation based on some available parameters (air temperature, relative humidity and sunshine duration) by using AI techniques. Additionally, it should be noted that many ANN applications in solar irradiation are oriented solely towards the use of software solutions, whereas the use of ANN hardware implementation is less common (Mellit, 2008; Mellit et al., 2008).

Therefore, in the present study we investigate the possibility of implementing MLP-predictors (MultiLayer Perceptron) into reconfigurable digital devices such as Field Programmable Gate Array (FPGA) in order to develop a hardware board which can be used for real-time prediction of solar irradiation in areas where there are no meteorological stations. This device (FPGA) allows the implementation of the neural network designs in parallel architectures in order to be used in real-time applications. FPGA technology is employed due to its development, flexibility and low cost. In fact, we have chosen the MLP because its architecture is not complicated and it can be implemented easily on FPGA than on any other ANN architectures or hybrid systems (ANFIS, GA-ANN, etc.) (Mellit, Shaari, & Mekki, 2008).

ANN is a parallel and distributed network of simple nonlinear processing units interconnected in a layered arrangement. Parallelism, modularity and dynamic adaptation are the three computational characteristics typically associated with ANNs. FPGA-based reconfigurable computing architectures are well suited to implement ANNs as can be exploited concurrently and rapidly to reconfigure and adapt the weights and topologies of an ANN. The hardware implementations are essential for applicability and for taking the advantage of ANN’s inherent parallelism. Compared to traditional microprocessor-based systems, fast growing FPGA technology offers a more powerful, efficient and flexible hardware platform including on-site (field-programmable) reconfiguration capability of hardware.

The main objective of this work is to develop and implement a suitable intelligent predictors for global solar irradiation in reconfigurable FPGA hardware Xilinx® Virtex-II FPGA (XC2v1000). Firstly, the different proposed ANN configurations \[ G = f(t, T, S, R_h), \] \[ G = f(t, T, S), \] \[ G = f(t, T, R_h), \] \[ G = f(t, S, R_h) \] \[ G = f(t, T) \] and \[ G = f(t, S) \] will be simulated under Matlab® and then written by using the Very High Description Language (VHDL). Once the best architectures are written in VHDL, the next step is to simulate and test these configurations under ModelSim®. The final step is to implement the best configurations into FPGA and develop the hardware boards. The implemented hardware MLP-predictors can present a real-time application for predicting and measuring the daily global

**Nomenclature**

ADR0, ADR1 address signals  
CE0, CE1 the control signals  
\( G \) global solar irradiation (Wh/m²/day)  
\( G_0 \) extraterrestrial solar irradiation (Wh/m²/day)  
\( K_I \) clearness index  
\( RH \) relative humidity (%)  
\( RH_{max} \) maximum relative humidity (%)  
\( S \) diffuse sunshine duration (h)  
\( SEL_0, SEL_1 \) signals for data selection  
\( S_0 \) extraterrestrial sunshine duration (h)  
\( S \) fraction of sunshine duration  
\( T \) air temperature (°C)  
\( T_{max} \) maximum temperature (°C)

**Abbreviations**  
CLB Configurable Logic Block  
FPGA Field Programmable Gate Array  
CLK Clock  
JOB Input Output Bonded  
JTAG Joint Test Action Group  
LC Logic Cell  
LUT Look-Up Table  
MBE Mean Bias Error  
MLP MultiLayer Perceptron  
MULT Multiplexer  
r correlation coefficient  
RMSE Root Mean Square Error  
VHDL Very High speed integrated circuit hardware Description Language
solar irradiation from other meteorological parameters such as air temperature, relative humidity and sunshine duration which are always available, since these are probably the most registered meteorological variables.

This paper is organized as follows: next section introduces the database used in this study; Section 3 provides the correlation between the solar irradiation and other meteorological data (air temperature, sunshine duration and relative humidity). MLP architectures used for predicting of the global solar irradiation as well as the developed architectures based on the VHDL are presented in Section 4, while Section 4.3.4, describes the methodology and the implementation of the different solar irradiation predictors. Section 5 gives the simulation results of the developed MLP models on Matlab®, ModelSim®, and FPGA for future experimental realization.

2. Available data description

The experimental data used in this work are the global solar irradiation, sunshine duration, air temperature and relative humidity. These data are available from 1998 to 2002 by the National Renewable Energy Laboratory (NREL) (1998). Fig. (1a) shows the daily evolution of global solar irradiation received on a horizontal surface and sunshine duration whereas Fig. (1b) shows the daily evolution of air temperature and relative humidity. These data correspond to Al-Madinah city (Latitude: 24.55 N° Longitude: 39.70 E°), Saudi Arabia. Fig. 2(a) illustrates the daily evolution of irradiation for the year 2002 at Al-Madinah site. It also shows the values of $G_0$, which represents the extraterrestrial radiation. Fig. 2(b) shows the daily evolution of sunshine duration for the year 2002. This figure shows clearly that there is seasonal trend, which is similar to that of the daily values of solar irradiation data. Corresponding curves of clearness indexes ($K_t = G/G_0$) values and sunshine duration fraction ($SS = S/S_0$) are presented in Fig. 2(c). Sunshine
duration is defined by WMO as the time during which the direct solar radiation is above 120 W/m².

The distribution of clearness index $K_t$ is around the yearly average clearness index of 0.7281. This shows that the global radiation at Al-Medina site is higher and many solar energy applications will have good prospects for success.

3. Correlation between solar irradiation, sunshine duration, air temperature and relative humidity

In the present section, we will present the different classical, linear and polynomial regression models between solar irradiation and sunshine duration, air temperature, and relative humidity. In addition, we will calculate the correlation coefficient between different regression models.

3.1. Correlation between global solar irradiation and sunshine duration

The relations between the fraction ($G/G_0$) and sunshine duration ($S/S_0$) are (Zekai, 2008):

$$\frac{G}{G_0} = a + b \left( \frac{S}{S_0} \right)$$  \hspace{1cm} (1)

$$\frac{G}{G_0} = a + b \left( \frac{S}{S_0} \right) + c \left( \frac{S}{S_0} \right)^2$$  \hspace{1cm} (2)

where $a$, $b$ and $c$ are the regression coefficients.

Fig. (3a) shows the correlation between daily global radiation and sunshine duration for Al-Madinah location. The correlation coefficient is 94%.

3.2. Correlation between global solar irradiation and air temperature

Fig. (3b) shows the correlation between the global solar irradiation and the air temperature at Al-Madinah. 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. The linear regression for the experimental data is given by:

$$\frac{G}{G_0} = a_1 + b_1 \left( \frac{T}{T_{\text{max}}} \right)$$  \hspace{1cm} (3)

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).

3.3. Correlation between global irradiation and relative humidity

Fig. (3c) shows the correlation between the global solar irradiation and the relative humidity at Al-Madinah site. The correlation model is given by the following formula:

$$\frac{G}{G_0} = a_2 + b_2 \left( \frac{RH}{RH_{\text{max}}} \right)$$  \hspace{1cm} (4)

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

4. Artificial neural network and FPGA

In this section, a brief introduction to artificial neural network, FPGA and the different steps for implementing the MultiLayer Perceptron on FPGA, which will be used in this work is presented.

4.1. Artificial neural network

ANNs have been used in many applications in science and engineering. An ANN is a parallel and distributed network of simple nonlinear processing units interconnected in a layered arrangement. Parallelism, modularity and dynamic adaptation are the three computational characteristics typically associated with ANNs (Zhu & Sutton, 2003).
Back-propagation (BP) has been widely adopted as a successful learning rule to find the appropriate values of the weights for NNs. 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 as shown in Fig. (4a). All these processing units carry out the same operation as shown in Fig. (4b), i.e., the sum of their weight inputs (see Eq. 5). Then they apply the result to a nonlinear function, named, activation function, which is generally based on the tangent sigmoid function (see Eq. 6) and is the most frequently used in the feed-forward type of ANN.

\[
y_j = f \left( \sum_i w_{ij} x_i + b_j \right)
\]

(5)

\[
f(y_j) = \frac{2}{1 + e^{-2y_j}} - 1
\]

(6)

where \(y_j\) is the output of the processing unit, \(w_{ij}\) the synaptic weight coefficient of the \(i\)th input of the processing unit, \(b_j\) is the bias.

The \textit{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, as will be described later.

4.2. Field Programmable Gate Array (FPGA)

FPGAs are a form of programmable logic (Amos, Omondi, & Aajapakse, 2006), which offer flexibility in design like software, but with performance speeds closer to Application-Specific Integrated Circuits (ASICs). With the ability to be reconfigured an endless number of times after having been manufactured, FPGAs have traditionally been used as a prototyping tool for hardware designers. However, as growing capacities of FPGAs have increased over the years, so has their use in reconfigurable computing applications.

The fundamental architecture of Xilinx FPGAs consists of a two-dimensional array of programmable logic blocks, referred to as Configurable Logic Blocks (CLBs). Fig. 5 shows the architecture of a CLB from the Xilinx Virtex-E family of FPGAs, which contains four logic cells (LCs) and is organized in two similar slices. Each LC includes a 4-input lookup table (LUT), dedicated to fast carry-look ahead logic for arithmetic functions, and a storage element (i.e., a flip-flop). A CLB from the Xilinx Virtex-II family of FPGAs, on the other hand, contains eight 4-input LUTs, and is over twice the amount of logic as a Virtex-E CLB. As we will see, the discrepancies in CLB architecture from one family to another are important factors to take into consideration when comparing the spatial requirements (in terms of CLBs) for circuit designs which have been implemented on different Xilinx FPGAs (Amos et al., 2006).

4.3. FPGA-based implementation of MLP

Neural network implementations on FPGAs have to deal with the usual issues of digital hardware implementations of neural network applications. But these issues become more acute because of
the specific constraints of FPGAs; this is particularly obvious when it is a matter of area consumption. Even if the software-like use of FPGAs is simple, the user should not forget that the compiler has to deal with these strong constraints. Nowadays, several techniques have been proposed for implementing the MLP on the FPGA; most of them can be found in Amos et al. (2006). The different steps for the implementation of the MLP on FPGA are presented in the following subsection.

4.3.1. Elementary neuron implementation

The neuron function consists in calculating the pondered sum function ($\sum w_i x_i$) and updating the state of the neuron by applying the activation function. The basic units of the neuron are:

- **Calculation unit of the pondered sum (MAC)**

Several types of adders can be used to achieve the states of the pondered sum: combinatorial, in series, dynamic, etc. (Pedroni, 2004). 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 presented in the literature. As an example, we can mention the possibility to use arithmetic units of calculation ‘series’. This technique needs registers of transformation series/parallel, which permits the elimination of the coding buses of information and therefore the clutter. Considering the number of calculations necessary for the simulation of the MLP network and the time granted to do these calculations, we did not consider it necessary to use this last technique. The multiplier of 18 bits optimized for the Xilinx bookstore proved to be quite sufficient (Mekki, Mellit, Salhi, & Khaled, 2008). Thus, this MAC (Multiply-Accumulate) is used which in our case has been realized by a multiplier of 18 bits associated to an adder smoothed on it in order to get an accumulation as shown in Fig. (6a).

- **Approximation unit for the activation function (AF)**

The activation function used is Tangent-Sigmoid (Eq. 7) 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 which uniform samples taken from the centre of a sigmoid function can be stored in a table for lookup. The regions outside the centre of the sigmoid function are still approximated in a linear segment.

In our study we have chosen the first approach, so in order to simplify the function expression, this was linearized on several intervals $[c_i, c_{i+1}]$ and its values are evaluated using two constants $(a_i$ and $b_i$) corresponding to this intervals (Amos et al., 2006; Mekki et al., 2008), so:

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

(7)

- **Storage unit of the weights and bias (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 (Pedroni, 2004). 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 in 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 $b$ (see Fig. (6b)).

where $i$ is the number of layers and $i=1$ is the input layer.

For $i > 1$:

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

(8)

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

By using the developed three units (MAC, AF and ROM unit), we can construct the elementary neuron which is shown in Fig. 7.

4.3.2. Techniques used in the numeric neural calculator

The neural networks are especially adapted to the parallel calculation. In the working phase, for a multilayer network, different possibilities of parallelism can be exploited. We present two of them:

- Parallelism in the layers.
- Parallelism in the calculation of the product and the sum of the product.
4.3.3. Pipeline technique

The inter-layers parallelism consists in exploiting the flood of data. The relaxation time is generally obtained by measuring the separating time of the presentation from a stimulus in input and the evaluation at the output of the network (i.e., the propagation time in the network). The calculations set is divided in different

Fig. 8. Parallelism of pipe-line.

Fig. 9. The different MLP-predictors: (a) \( G - \tilde{f}(t, S, T, RH) \), (b) \( G - \tilde{f}(t, T, S) \), (c) \( G - \tilde{f}(t, T, RH) \), (d) \( G - \tilde{f}(t, S, RH) \), (e) \( G - \tilde{f}(T) \), and (f) \( G - \tilde{f}(S) \).
modules of treatment (layers) put tip-to-tip. This technique, named ‘pipe-line’, permits the decrease of length of the calculation cycle (Pedroni, 2004).

In the case of neural networks, each module can be made up of one or several layers of the network and the objective is to separate the calculation quantity in autonomous tasks, so that the computing time of each task is less than the total cycle time.

Fig. 8 shows the pipe-line technique for a network of three layers with a separation following every layer. It represents the three steps of time and propagation of the calculations corresponding to different stimuli.

4.3.4. Coding data

According to the literature research on the implementation of the different types of numeric applications in VLSI technology, it was noted that the representation in fixed-point is the more widely used because it is very easy to be implemented than the representation in floating-point and most digital computers support only this type of representation (Nelson & Marcelo, 2002).

In our study, a representation in fixed-point on 18 bits is chosen (9 bits for the integer part and 9 bits for real part), because the FPGA circuits used in this study contain multipliers of 18 bits, which facilitate the realization of the MAC (multiply-accumulate) that represents the main element of the neuron (Mekki et al., 2008).

5. Methodology and implementation

In this section, the different steps for implementing an intelligent predictor for solar irradiation from other meteorological parameters are presented. The procedure can be summarized as follows:

1. Define the inputs and the output of the employed MLP for different proposed configurations: \( G = f(t, T, S, RH) \), \( G = f(t, T, S) \), \( G = f(t, T, RH) \), \( G = f(t, S, RH) \). 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 consisting of:
   (a) Mapping: conversion of the VHDL representations into the structures which the target device will consist of.
   (b) Placement: the mapped logic blocks are settled into specific configurable logic blocks of the target device.
   (c) Routing: connection between the selected configurable logic blocks by means of routing resources provided by the adapted FPGA Xilinx® Virtex-II (XC2v1000) device.
   (d) Back annotation: a representation of the place and routed design is produced including timing data for logic and routing delays.
7. Download the bit-stream file into the target device to make it operative.

5.1. 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) \( G = \tilde{f}(t, T, S, RH) \) 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 Fig. 9(a).

(2) \( G = \tilde{f}(t, T, S) \) has as input parameters the air temperature, sunshine duration, and day of year, while the global solar irradiation in the output, shown in Fig. 9(b).

(3) \( G = \tilde{f}(t, T, RH) \) has as input parameters the air temperature, relative humidity, and day of year, while the output is the global solar irradiation, shown in Fig. 9(c).

(4) \( G = \tilde{f}(t, RH, S) \) has as input parameters the relative humidity, sunshine duration and day of year, while the output is the global solar irradiation, shown in Fig. 9(d).

(5) \( G = \tilde{f}(t, T) \) has as input parameters the air temperature and day of year, while the output is the global solar irradiation, shown in Fig. 9(e).

(6) \( G = \tilde{f}(t, S) \) has as input parameters only the sunshine duration and the day of year while the output is the global solar irradiation, shown in Fig. 9(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). A database of 1825 patterns is divided into two parts, a dataset of 1460 (4-years) patterns used for training the different MLP-predictors (configuration) and a dataset of 365 (1-year) patterns used for testing and validating each configuration. A soft computing program has been implemented for the prediction of the global solar irradiation based on the Levenberg–Marquardt (LM) algorithm (Haykin, 1999); Matlab® Ver. 7.5 is used in this simulation.

### 5.2. FPGA-based implementation

Before starting the implementation of the developed best configurations (MLP-predictors) on FPGA, the elementary neuron, which can be used for construction of the network (MLP) is initially presented. The mathematical function expressed by a neuron must be formulated by simple elementary calculations (addition and multiplication), which allow the possibility to synthesize the neuron by the VHDL.

The structural description of the neuron permits to specify some features, as the number of inputs and outputs, the activation function, and the size of the binary words (weight, word memory, etc.). As shown above, the elementary neuron is composed of three main elements which are: the MAC, the unit which allows the approximation of the activation function (AF), and the unit of the storage of weights and bias (ROM).

As presented in Section 4.3, the MAC is the unit of the calculation which allows the calculation of the pondered

### Table 1

<table>
<thead>
<tr>
<th>MLP-configuration</th>
<th>Architecture: input/hidden/output</th>
<th>RMSE</th>
<th>MBE</th>
<th>( r )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) ( G = \tilde{f}(t, T, S, RH) )</td>
<td>4 ( \times ) 4 ( \times ) 1</td>
<td>0.00110</td>
<td>4.836</td>
<td>0.9720</td>
</tr>
<tr>
<td>(2) ( G = \tilde{f}(t, T, S) )</td>
<td>3 ( \times ) 4 ( \times ) 1</td>
<td>0.00110</td>
<td>4.622</td>
<td>0.9744</td>
</tr>
<tr>
<td>(3) ( G = \tilde{f}(t, T, RH) )</td>
<td>3 ( \times ) 4 ( \times ) 1</td>
<td>0.00290</td>
<td>6.448</td>
<td>0.8978</td>
</tr>
<tr>
<td>(4) ( G = \tilde{f}(t, RH, S) )</td>
<td>3 ( \times ) 4 ( \times ) 1</td>
<td>0.00120</td>
<td>4.717</td>
<td>0.9731</td>
</tr>
<tr>
<td>(5) ( G = \tilde{f}(t, T) )</td>
<td>2 ( \times ) 4 ( \times ) 1</td>
<td>0.00414</td>
<td>6.3430</td>
<td>0.8927</td>
</tr>
<tr>
<td>(6) ( G = \tilde{f}(t, S) )</td>
<td>2 ( \times ) 4 ( \times ) 1</td>
<td>0.00127</td>
<td>4.7156</td>
<td>0.9724</td>
</tr>
</tbody>
</table>

Fig. 11. The ROM simulation in ModelSim.

Fig. 12. Parallel implementation of a neural network.
sum, therefore its description in VHDL consists in specifying the number of inputs, their sizes, as well as the operations necessary for the accumulation and multiplication (addition and multiplication).

Fig. 13. Comparison between measured and predicted global solar radiation data for different configurations: (a) $G = f(t, T, S, RH)$, (b) $G = f(t, S, RH)$, (c) $G = f(t, T, RH)$, (d) $G = f(t, S, RH)$, (e) $G = f(t, T)$, and (f) $G = f(t, S)$. 
Fig. 14. Cumulative distribution function between measured and predicted global solar irradiation data for different configurations: (a) $G \sim \tilde{f}(t, S, R, H)$, (b) $G \sim \tilde{f}(t, T, S)$, (c) $G \sim \tilde{f}(t, T, R, H)$, (d) $G \sim \tilde{f}(t, S, R, H)$, (e) $G \sim \tilde{f}(t, T)$, and (f) $G \sim \tilde{f}(t, S)$.
The description in VHDL of the approximation unit of the activation function serves to find the approximation intervals, and the constants \(a_i\) and \(b_i\) which describe the linear segments defined as 

\[Y_i = a_i \times X + b_i.\]

The structural description of a neuron with VHDL allows during the compiling step to specify generally some characteristics like the input numbers, the data size and even to modify the type of some arithmetic operators such as the adder or the multiplier the input numbers, the data size and even to modify the type of the compiling step to specify generically some characteristics like ModelSim the constants of the neuron module is presented in Appendix A.

Concerning the description of the ROM in the VHDL we are noted from Table 1 that one hidden layer within four neurons is sufficient for predicting the global solar irradiation data; the simplest 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 study (see Fig. 12).

The hardware implementation of the MLP on FPGA is based on two main elements (Mekki et al., 2008):

- 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.

### 6. Results and discussion

This section is subdivided into three subsections: the first one concerns with the MatLab simulation results while the second section deals with the results obtained by using the VHDL as well as the simulation results on ModelSim. The FPGA-implementation of the developed models (MLP-predictors) is presented in the third subsection.

#### 6.1. MatLab-simulation results

For several simulations, the best performance for each configuration as well as the statistical test between measured and predicted global solar irradiation is illustrated in Table 1. Fig. 13 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. The rest of the configurations (the third and fifth) cannot provide very acceptable results. It should be noted from Table 1 that 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. Fig. 14 illustrates the cumulative distribution function between measured and predicted data. The correlation coefficient obtained is between 89% and 97%, and the MBE varied between 4% and 6%. We can conclude that the sunshine duration increases clearly the performance of the predictor, however, the relative humidity cannot influence the accuracy. When only one parameter is used as input it is clearly proven that the sixth model which has as input only the sunshine duration provides much better results than the fifth predictor which has as input only the air temperature. 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

#### Table 2

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Accuracy (r)</th>
<th>Simplest architecture</th>
<th>Availability, cost and efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) (G = f(t, T, S, RH))</td>
<td>0.9720</td>
<td>(4 \times 4 \times 1)</td>
<td>Expensive (sunshine duration sensor), not always available and simplest architecture</td>
</tr>
<tr>
<td>(2) (G = f(t, T, S))</td>
<td>0.9749</td>
<td>(3 \times 4 \times 1)</td>
<td>Expensive (Sunshine duration sensor), not always available, accurate and simplest architecture</td>
</tr>
<tr>
<td>(3) (G = f(t, T, RH))</td>
<td>0.8978</td>
<td>(3 \times 4 \times 1)</td>
<td>Not expensive, sensors always available, not accurate, and simplest architecture</td>
</tr>
<tr>
<td>(4) (G = f(t, S, RH))</td>
<td>0.9730</td>
<td>(3 \times 4 \times 1)</td>
<td>Expensive (sunshine duration sensor), not always available, accurate and simplest architecture</td>
</tr>
<tr>
<td>(5) (G = f(t, T))</td>
<td>0.8927</td>
<td>(2 \times 4 \times 1)</td>
<td>Not expensive, sensor always available, not accurate, and simplest architecture</td>
</tr>
<tr>
<td>(6) (G = f(t, S))</td>
<td>0.9724</td>
<td>(2 \times 4 \times 1)</td>
<td>Expensive (Sunshine duration sensor), not always available, accurate and simplest architecture</td>
</tr>
</tbody>
</table>
for sunshine duration is not always available, especially in under-
development countries (Mellit et al., 2008) 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:

\[
\hat{y} = \sum_{k=1}^{M} \left( \frac{1 + \exp\left( - \sum_{j=1}^{N} w_1(i,j)x(l) + b_1 \right)}{1 + \sum_{j=1}^{N} w_1(i,j)x(l) + b_1} \right) w_2(k)b_2(k)
\]

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). \( \hat{y} \) is the predicted global solar irradiation, \( M \) and \( N \) are the number of neurons in the hidden layer and in the input layer, respectively. The experimental values of \( w_1, w_2, b_1 \) and \( b_2 \) for both MLP-predictors can be seen in Appendix B.

6.2. VHDL-simulation results

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. The ISE Ver. 10.1 of Xilinx is used in the present study for developing the VHDL source code.

The choice of VHDL is justified by its flexibility and the fact that it is a standard language (IEEE 1076). Furthermore, it is a target device and independent designing tool, i.e., with one design description and many device architectures can be included.

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_vector type, 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 numeric architectures of these predictors are shown in Fig. 15(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 is presented in Fig. 16(a) and (b), 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 Fig. 17(a) and (b) for the designed MLP-predictors \(G = f(t, T, S)\) and \(G = f(t, T, RH)\). 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.

6.3. FPGA-implementation of MLP-predictors

The implementation of the designed MLP-predictors \(G = f(t, T, S)\) and \(G = f(t, T, RH)\) was achieved using a Xilinx® Virtex-II FPGA (XC2v1000) from Xilinx (see Fig. 18). This FPGA has 3072 slices and 6114 logic cells. The FPGA has also thirty-two 18 × 18 multipliers, as well as thirty-two 18 K bit modules of dedicated dual-port RAM. Fig. 19 shows the Register Transfer Level (RTL) schematic of both MLP-predictors. Table 3 illustrates the FPGA logic resources used to develop the predictors addressed in this paper. Therefore, once the bit-stream files for both MLP-predictors are downloaded into the target device to make it operative, the final step is to develop the experimental realization of the electronics hardware circuit. A simplified schematic of this electronics board is shown in Fig. 20.

The designed circuit contains: two sensors for the sunshine duration and air temperature \(G = f(t, T, S)\), or relative humidity and air temperature \(G = f(t, T, RH)\), conditioning circuits, A/D
converters, Joint Test Action Group (JTAG) and FPGA device. The proposed configuration will be developed for the second part of this paper (Experimental validation).

7. Conclusion and future actions

In this paper, intelligent predictors for solar irradiation are proposed and implemented into reconfigurable FPGA Virtex-II (XC2v1000). 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.

![Graphs](image-url)

Fig. 17. Comparison between Matlab® and ModelSim® results: (a) \( G = \tilde{f}(t, T, S) \) and (b) \( G = \tilde{f}(t, T, RH) \).
The designed predictors have huge potential in industrial electronics for developing intelligent sensors for solar irradiation based on the FPGA hardware.

Our future actions can be summarized as follows:

- Experimental realization of the developed predictors on FPGA for real-time measuring of solar irradiation (electronics board) by taking in account the expenditure required.
- Improve the accuracy of the designed predictors by using a large dataset and introducing some geographical parameters.
- Modification of the prediction times at 1 h, 10 min, etc.

The used Virtex-II FPGA (XC2V1000) development board.

### Table 3

Synthesis report of the implemented MLP-predictors.

<table>
<thead>
<tr>
<th>Logic utilization</th>
<th>Used</th>
<th>Available</th>
<th>Utilization (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of slices</td>
<td>1331</td>
<td>5120</td>
<td>25</td>
</tr>
<tr>
<td>Number of slice flip-flops</td>
<td>404</td>
<td>10240</td>
<td>3</td>
</tr>
<tr>
<td>Number of 4 input LUTs</td>
<td>2374</td>
<td>10240</td>
<td>23</td>
</tr>
<tr>
<td>Number of bonded IOBs</td>
<td>75</td>
<td>324</td>
<td>23</td>
</tr>
<tr>
<td>Number of MULT18X18s</td>
<td>17</td>
<td>40</td>
<td>42</td>
</tr>
<tr>
<td>Number of CLKs</td>
<td>2</td>
<td>16</td>
<td>12</td>
</tr>
</tbody>
</table>
Acknowledgment

The first author would like to thank the International Centre for Theoretical Physics (ICTP), Trieste (Italy) for providing the materials and the computers facilities for performing the present work. This work was also supported by the Ministry of Higher Education & Scientific Research (Algiers) under project number: J0201720008012.

Appendix A. The VHDL code source of the neuron module

```
Library declaration
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_SIGNED.ALL;

Entity declaration of the neuron
Entity Neurone_1_X is
  Generic
  (constant W0:integer:=10;
   constant W1:integer:=20;
   constant W2:integer:=20;
   constant B: integer:=20);
  Port (X: in std_logic_vector (17 downto 0);
       adr: in std_logic_vector (1 downto 0);
       Ck: in std_logic; Ce: in std_logic; Load: in std_logic;
       En_S: in std_logic;
       S: out std_logic_vector (17 downto 0));
End Neurone_1_X;

Architecture of the neuron
architecture Behavioral of Neurone_1_X is
  Component declaration of the ROM
  component ROM_3_X is
    generic (constant x0:integer:=10;
             constant x1:integer:=20;
             constant x2:integer:=20;
             constant Size: Natural:=32);
    port (adr: in std_logic_vector (1 downto 0);
          d: out std_logic_vector (Size-1 downto 0));
  end component ROM_3_X;

  MAC declaration
  Component MAC is
    Port (X: in std_logic_vector (17 downto 0);
          Y: in std_logic_vector (17 downto 0);
          Z: in std_logic_vector (17 downto 0);
          A: out std_logic_vector (35 downto 0);
          CK: in std_logic; Ce: in std_logic;
          LOAD: in std_logic);
  end component;

  Component declaration of activation function
  component seg_function is
    Port (X: in std_logic_vector (17 downto 0);
          Y: out std_logic_vector (17 downto 0);
          en_seg: in std_logic);
  end component;

  Signal W:std_logic_vector (17 downto 0);
  Signal A:std_logic_vector (17 downto 0);
  Signal Z:std_logic_vector (17 downto 0);
  Signal Y:std_logic_vector (17 downto 0);
```

Fig. 20. Simplified schematic of the proposed electronics board.
Appendix B. Weights and bias

The weights and bias for the second MLP-predictor $G = \hat{f}(t, T, S)$ (one hidden layer within four neurons):

$$
\begin{align*}
W_1 &= \begin{bmatrix} -1.5150 & 0.2181 & 5.0313 \\ -1.5028 & 0.2685 & 5.0393 \\ -0.0553 & 1.6722 & -0.0926 \\ 5.0221 & 18.137 & 191.41 \end{bmatrix}, \\
W_2 &= \begin{bmatrix} 15.2070 \\ -15.1699 \\ 69.9007 \\ -0.0255 \end{bmatrix}, \\
b_1 &= \begin{bmatrix} -1.9540 \\ -2.0199 \\ -4.0259 \\ -14.7969 \end{bmatrix}, \quad \text{and} \quad b_2 = [69.8752] \end{align*}
$$

The weights and bias for the third MLP-predictor $G = \hat{f}(t, T, RH)$ (one hidden layer within four neurons):

$$
\begin{align*}
W_1 &= \begin{bmatrix} -0.1005 & -0.0735 & 3.5152 \\ 0.0629 & 0.0236 & -3.4991 \\ -179.8510 & -9.3032 & 91.2333 \\ -0.5042 & -6.2012 & 1.5681 \end{bmatrix}, \\
W_2 &= \begin{bmatrix} 18.2444 \\ 18.2686 \\ 0.0338 \\ 0.2029 \end{bmatrix}, \\
b_1 &= \begin{bmatrix} 1.6246 \\ 122.8772 \\ 4.7203 \end{bmatrix}, \quad b_2 = [0.0924] \end{align*}
$$

References


