Neural Comput & Applic (2019) 31:2469–2483
DOI 10.1007/s00521-017-3207-1

ORIGINAL ARTICLE

Fuzzy logic control on FPGA for two axes solar tracking

J. de la Cruz-Alejo[1] · R. Antonio-Méndez[1] · M. Salazar-Pereyra[1]

Received: 9 September 2015 / Accepted: 4 October 2017 / Published online: 24 October 2017

Abstract This paper presents a practical fuzzy controller two axes solar tracking-based realization on digital FPGA hardware. The fuzzy logic control is based according to Mamdani rules, alpha levels, max–min operations and defuzzification method. Operations and algorithms are reduced using look-up tables for the membership values which are stored as digital values and accessed to the control process. The feasibility and versatility of the proposed technique as well as its potential as a low-cost design for solar tracking control on digital field-programmable gate array (FPGA) are shown by simulated and experimental results in a photovoltaic system under different operation conditions. The proposed realization exhibits good performance related to the control and efficiency.

Keywords Sun tracking · Fuzzy logic control · Alpha levels · FPGA · Mamdani rules · max–min · Look-up tables · Photovoltaic system (PV)

✉ M. Salazar-Pereyra
msalazar@tese.edu.mx

J. de la Cruz-Alejo
jdelacruz@tese.edu.mx

R. Antonio-Méndez
ric_xof@hotmail.com

[1] Mechanical and Mechatronic Department, Tecnológico de Estudios Superiores de Ecatepec, Av. Tecnológico, Esq. Av. Hank González, Col. Valle de Anáhuac, 55210 Ecatepec, Estado de México, Mexico

1 Introduction

Solar energy received every 10 days on Earth equals all the known reserves of oil, coal and gas. Mexico receives high-quality solar power in more than half of its territory: $G = 1000$ W/m^2 on average in states of high insolation. The use of this source of useful energy aims generally detach from the continued use of fossil fuels, allowing savings in non-renewable energy and in addition to amortize the environmental impact [1–3]. In the other hand, the quantity of electricity that can be generated by a solar PV system depends on a number of factors, including the solar resource available at the installation location measured as the insolation, ambient conditions such as temperature, wind speed, dust and cloud cover, spectral distribution of incident radiation, angle of incidence of solar radiation and operational efficiencies of system components [4–7]. So, in PV systems, one of the most important decisions is to select the best control technique suited to the requirements and feasibility of the proposed design approach. The decision of this logic element is high enough for serving as a good demonstrator that can be implemented by a selected hardware [8, 9]. The implementation can be done using general-purpose processors which depend fully on software for the realization or adapting a general-purpose processor to perform dedicated fuzzy instructions. The approach is a trade-off between speed and complexity. An alternative is using an exclusive hardware to perform the fuzzy operations as a closely related approach through dedicated fuzzy circuits or application-specific integrated circuits (ASICs). The approach leads to relatively high-speed operation, but is more costly. FPGAs are hardware devices used as user-programmable ASICs. These devices have the availability of software tools to generate efficient and flexible hardware description, also brings easiness to the reconfiguration

process. Moreover, FPGA designs can already be modeled, simulated and verified [10].

In the other hand, novel fuzzy control methods have been developed and implemented as a programming platform to control different processes in different areas [11]. So, one of the mathematical disciplines with the highest number of followers is the fuzzy logic technique, which is the logic that uses expressions that are neither completely true nor completely false, i.e., is the logic applied to concepts that can take any truth value in a set of values ranging between two extremes, absolute truth and complete falsity [9, 12–16]. Due to their heuristic nature associated with simplicity and effectiveness, for both linear and nonlinear systems, the fuzzy logic controller (FLC) has showed their outstanding features in implementations for solar tracking systems [17–19]. Monitoring tracking systems can be classified based on their movement. This can be a single axis or two axes. In the case of a single-axis mode, the motion can be in various ways: east–west, north–south or parallel to the earth's axis [20–22].

The main purpose of this work is to implement and demonstrate a practical solar tracking on digital FPGA hardware, reducing operations and hardware in each stage of the FLC which is designed based on *Mamdani rules*, *max–min* operations and alpha levels for the defuzzification, and Sect. 2 describes the sun tracking system and stages of the fuzzy logic control that involve the parameters and characteristics that would guarantee the correct operation of this control technique for its implementation on FPGA. In Sect. 3, a comparison between experimental results obtained and simulated using MATLAB is made. Conclusions are presented in Sect. 4.

2 Sun tracking system

The sun tracking system is implemented using sensors of sun intensity which are the input to the FLC which is designed and implemented on FPGA using Mamdani method and actuators which consist of two servomotors that perform the control action. Figure 1a, b shows the block diagram of the overall system. Data acquisition is carried out by four ambient light sensors which convert light intensity into digital output values. Two of these sensors (S1 and S2) are used to obtain the digital output values labeled as *Ds1* and *Ds2*. Then, with these values through mathematical operations, the error *e1* and derivative of the error *de1* are obtained. These sensors are used to determine the orientation for solar altitude angle. Same manner, the sensors S3 and S4 are used to obtain the digital output values labeled as *Ds3* and *Ds4,* used to obtain the error *e2* and derivative of the error *de2,* which determine the orientation for solar azimuth angle. Figure 1b shows in more detail the internal structure of the acquisition block. Signals *e1, e2, de1* and *de2* are in turn inputs to the fuzzy logic controller, where k is the sampling time, $e(k)$ represents the present value of the error and $e(k-1)$ the last value of the error. These values are stored in internal registers which are accessed via SPI (Serial Peripheral Interface). SPI serves as interface from data acquisition to the FLC. The FLC block contains the programmed syntaxes in VHDL code for the stages of the FLC. The Outputs of the FLC are the control actions for the servomotors. These values are sent to servomotors through the RS-232 protocol in digital mode. This protocol contains the commands for positioning according to the own configuration of each servomotor, which performs the control action for the azimuth and altitude angles. Also, this protocol realizes the synchronizing between data output of the FLC and the rotation mechanisms of the servomotors. Finally, Fig. 2 shows the internal blocks of the FLC that consist of fuzzification, inference, rules and defuzzification stages which in turn perform the control actions for the solar rotation mechanism on two axes. One of them rotates in the solar altitude angle and the other in the solar azimuth angle. So, the novelty of the paper is to implement a control using fuzzy logic techniques in hardware, on FPGA platform in particular. Also, operations in the controller process are reduced using look-up tables and alpha level in the controller stages. For example, operations as described by Eq. 1 is solved using a spreadsheet on a pc whose resultant values are captured in look-up tables. These values are

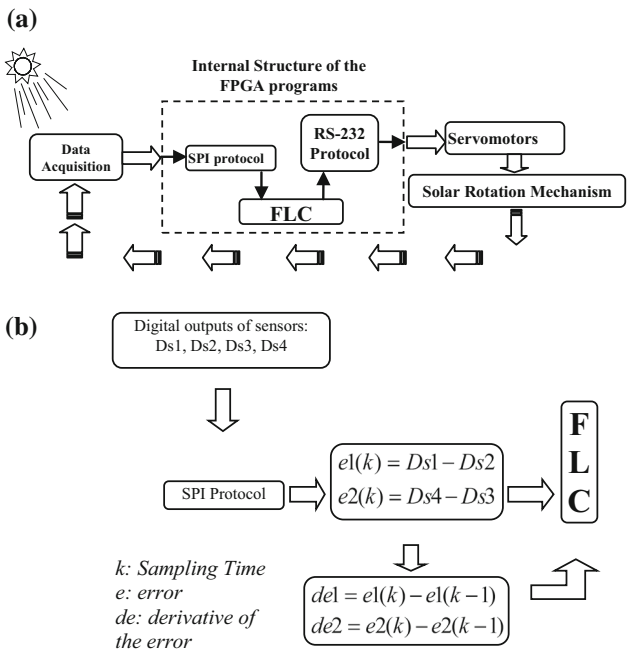

Fig. 1 System diagram. **a** Diagram for the sun tracking system. **b** Data acquisition for de FLC

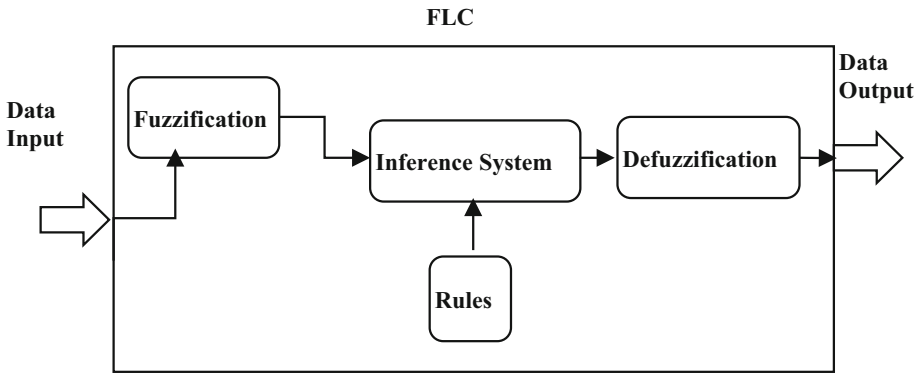

Fig. 2 Fuzzy logic control (FLC)

accessed in subsequent stages involving the control implementation on FPGA. This reduces the operations that resolve the FLC on the FPGA. Another advantage using fuzzy techniques for a solar tracking control is that it does not require a previous modeling, only the experience of an operator. In this case, the inputs were obtained using solar sensors and the outputs were obtained through Eqs. (1–7), for the azimuth and altitude angles.

2.1 Solar radiation

In order to estimate the solar radiation potential for the particular case in Mexico City, we have [23]:

The equation time is given by

$$\mathrm{ET} = 9.87\sin(2B) - 7.53\cos(B) - 1.5\sin(B)[\mathrm{min}]\ldots \tag{1}$$

where $B = (N-81)\frac{360}{364}$

The general equation to calculate the apparent solar time (AST) is:

$$\mathrm{AST} = \mathrm{LST} + \mathrm{ET} \pm 4(\mathrm{SL} - \mathrm{LL}) - \mathrm{DS}, \tag{2}$$

where LST = official local time or standard, ET = equation of time, SL = standard length, LL = local length, DS = daylight saving.

Decline is given by:

$$\delta = 23.45\sin\left[\frac{360}{365}(284+N)\right] \tag{3}$$

The decline can be given in radians:

$$\begin{aligned}\delta = {} & 0.006918 - 0.399912\cos(\tau) + 0.70257\sin(\tau) \\ & - 0.006758\cos(2\tau) + 0.000907\sin(2\tau) \\ & - 0.002697\cos(3\tau) + 0.00148\sin(3\tau),\end{aligned} \tag{4}$$

where τ is called the daily angle, given (in radians) by:

$$\tau = \frac{2\pi(N-1)}{365}$$

Hour angle

$$h = \pm 25(\text{number of minutes from local solar noon}) \tag{5}$$

$$\mathrm{LST} = 12 - \mathrm{ET} \pm 4(\mathrm{SL} - \mathrm{LL}) \tag{6}$$

$$\begin{aligned}\text{Angle of incidence, } \theta\cos(\theta) = {} & \sin(L)\sin(\delta)\cos(\beta) \\ & - \cos(L)\sin(\delta)\sin(\beta)\cos(Z_S) \\ & + \cos(L)\cos(\delta)\cos(h)\cos(\beta) \\ & + \sin(L)\cos(\delta)\cos(h)\sin(B)(Z_s) \\ & + \cos(\delta)\sin(h)\sin(\beta)\sin(Z_s),\end{aligned} \tag{7}$$

where β = Solar tilt angle.

The servomotors used as actuators were the *AX-12* which have a resolution of 0.35°, and the operating range for the position is limited to 0–300°, for each one, as shown in Fig. 3. This range is represented by digital values from 0 to 1023. Also, servomotors are able to send parameters in packets like temperature at which they are operating like torque and movement speed through the RS-232 or RS-245 protocol with a transfer rate up to 1Mbps. It can operate at a supply voltage from 7 to 10 V.

The communication protocol is controlled by the main controller (FLC) which synchronizes the transmission speed, the structure of the sending packets and the state of received packets according to the transmitted command. After sending the instruction to the servomotor, this returns either a status packet error or the corresponding requested action. One advantage of this type of servomotors lies in individual identifiers that can be set for each servomotor. With this, everyone can operate with the same protocol in continuous sequence from the main control defined in the package to the individual identifier for which the action command is assigned, as shown in Fig. 4.

The instruction packet structure is as follows:

$$|0\times FF|\ |0\times FF|\ |ID|\ |Length|\ |Instruction|\ |Parameter\,1| \cdots |Parameter\,N|\ \ |Check\ Sum|,$$

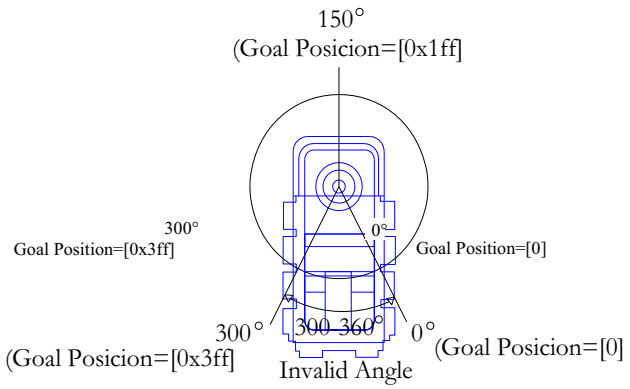

Fig. 3 Range of motion of the servomotor AX-12

where

- The first two bytes indicate to the actuator that a package is on the way.
- ID: The number that identifies the actuator can take a range of values from 0 to 254.
- Length: The length of the packet is given by "Number of parameters + 2"
- Instruction: The statement that depends on the type of parameter to be sent can be read or written.
- Parameter 1 … N is used if the instruction requires additional information in the package.
- Check Sum: This value is calculated by the following equation:

$$\text{Chek Sum} = \sim(\text{ID} + \text{Length} + \text{Instruction} + \text{Parameter 1} + \cdots \text{Parameter N}) \quad (8)$$

If the calculated Check Sum value is greater than 255, then the two least significant bytes are taken to set this value. For the communication to the FPGA, the same packet structure is implemented using the RS-232 protocol. Also, the timing for sending the control action dictated by the fuzzy controller is achieved. The data sequence sent is in packets of 8 bits, which are required to implement the RS-232 protocol in the FPGA. These packages are pre-calculated and accessed via look-up tables. The schematic blocks for the acquisition data of the sun tracking system implemented on FPGA is shown in Fig. 5.

2.2 Fuzzification

We define the input error e_i (*representing e1 or e2*) by five fuzzy variables e_i ($i = 1, 2, 3, 4, 5$), which conform to the linguistic variables, labeled PB (positive big), PS (positive small), ZO (zero), NS (negative small) and NB (negative big). Also, for the derivative error, we define the input de_i (*representing de1 or de2*) by five fuzzy variables de_j ($j = 1,2,3,4,5$), which conform to the linguistic variables, labeled PB (positive big), PS (positive small), ZO (zero), NS (negative small) and NB (negative big). The output fuzzy variable or the control quantity also uses five fuzzy variables which are described by fuzzy linguistic control quantities Ax_k ($k = 1, 2, 3, 4, 5$), partitioned on the control universe from 0 to 1023, which is proposed due to the movement of the servomotors because they have 1024 possible values with a range of motion from 0° to 300°, where 0° is represented by a digital value of 0° and 300° by 1023. Figure 6 shows the membership functions in the universe of discourse for the input and output variables.

In this case, because the sensors have a resolution of 8 bits ($2^n - 1$), the universe of discourse for the two input variables (*e1, de1*) is partitioned from − 127 to 127, which represents the ranges $-150° \leq e \leq 150°$ and $-150° \leq de \leq 150°$, where e is measured in degrees and de is measured in degrees per second. So, the maximum value $e1(k) = Ds2 - Ds1$ can be either negative or positive. Depending on whether *Ds2* is greater than *Ds1,* the result is a positive number, otherwise if *Ds1* is greater than *Ds2*, the result is a negative value.

The output of the sensors to detect or not lighting, are in turn, the data used to determine the error and derivative of the error as indicated in Fig. 1b. The error is obtained by:

$$e_1(k) = D_{s2}(k) - D_{s1}(k) \quad (9)$$

This manner, assuming that at a certain sampling time k, the sensor 2 has a value $Ds2 = 0$ and the sensor 1, $Ds1 = 127$, and substituting in (9),

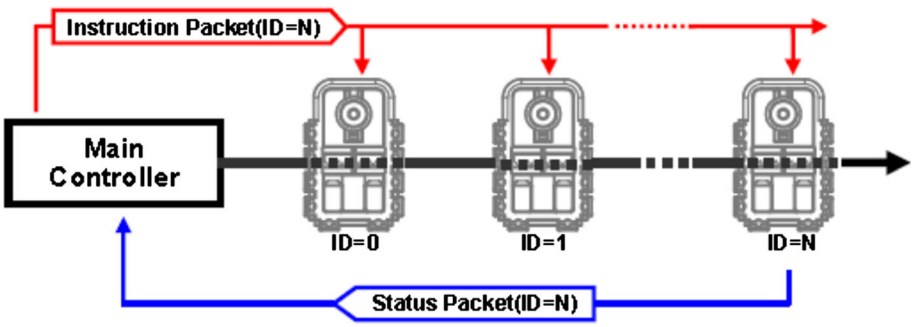

Fig. 4 Communication between devices

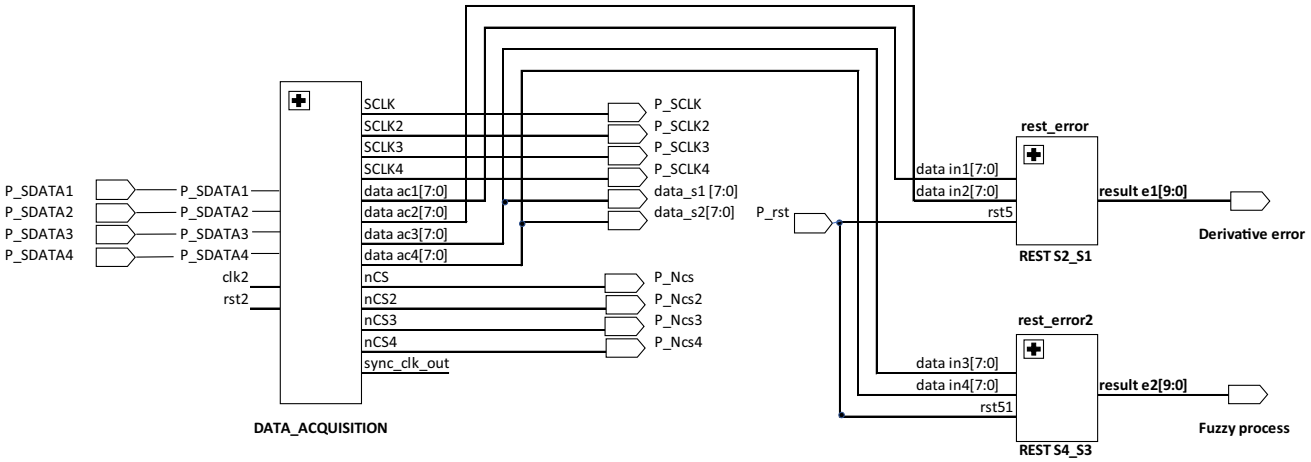

Fig. 5 Schematic blocks for the acquisition data

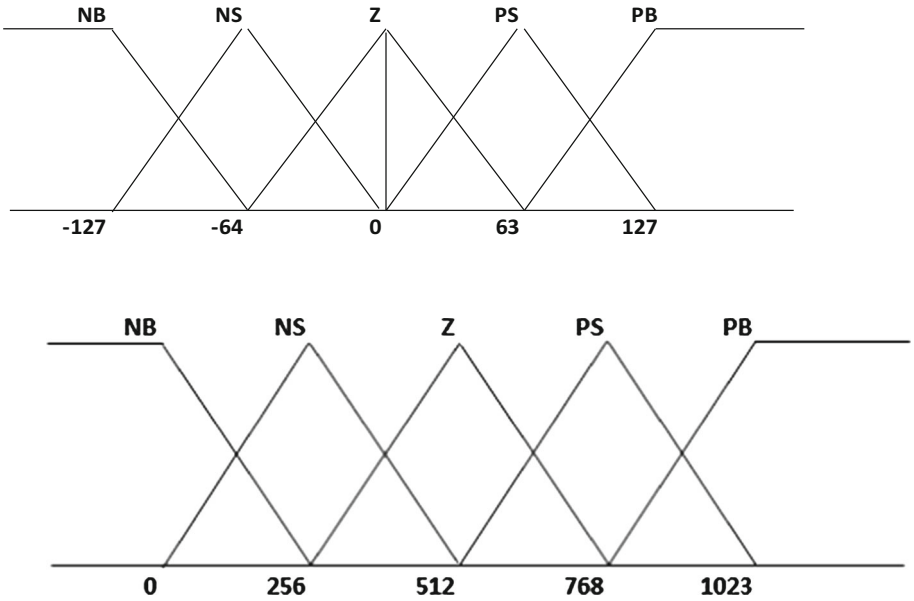

Fig. 6 Membership functions for input and output variables

$e_1(k) = 0 - 127 = -127$. Now, for the same sampling time k, the sensor 2 has a value $Ds2 = 127$ and the sensor 1, $Ds1 = 0$, substituting in (9), $e_1(k) = 127 - 0 = +127$. Hence, the range is selected for the universe of discourse for the input variables *e1* and *e2*.

For the derivative of the error, it is defined by:

$$de_1 = e_1(k) - e_1(k-1) \tag{10}$$

The same approach is supposed to define its universe of discourse. This is, if at any sampling time k, the current error has a value $e(k) = 0$ and $e(k-1) = 127$, substituting in (10), $de_1 = 0 - 127 = -127$. In the case, if $e(k) = 127$ and $e\ (k-1) = 0$, substituting in (10), $de_1 = 127 - 0 = +127$. This defines the range of the second input variable to the controller, which is *de1* and *de2*.

For the output variables *Ax1* and *Ax2*, the same linguistic variables are adopted, but with a different range for each fuzzy set as defined below:

- NB Negative big represents the fuzzy set in the range [0, 256]
- NS Negative small represents the fuzzy set in the range [0, 512]
- Z Zero represents the central fuzzy set within the interval [256, 768]
- PS Positive small represents the fuzzy set within the interval [512, 1023]
- PB Positive big represents the fuzzy set on the interval [768, 1023]

The membership function chosen for the FLC is triangular as shown in Fig. 6, which is given by the parameters a, b, c, d as follows:

$$\text{triangle}(x;a,b,c) = \begin{cases} 0, & x \le a \\ \dfrac{x-a}{b-a}, & a \le x \le b \\ \dfrac{c-x}{c-b}, & b \le x \le c \\ 0, & c \le x \end{cases} \quad (11)$$

Using *max–min* operators, (11) can be represented as:

$$\text{triangle}(x;a,b,c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), \ 0\right) \quad (12)$$

Once the membership function is selected with the required parameters, the values of membership function for each fuzzy set are obtained. For this, alpha-levels method is used, which is based according to Fig. 7, where $\mu(x)$ is the membership value reached for each fuzzy set in the X universe for any input. Each alpha level consists of x_0 and x_f which are the initial point value (alpha-level initial) and the final point value (alpha-level final) [24].

Table 1 shows the fuzzy sets for each input *e1* and *de1*, where *Wnx* and *Wny* are the weight vectors containing the membership functions in the universe of discussion for both inputs. Figure 8 shows the flowchart for the fuzzy process, where an iterative cycle is used to find and obtain the membership values by increasing a variable (i) from zero to n (n is the number of bits used to represent all membership values; in this case, from 0 to 255 for each fuzzy set, this is represented by an array of 256 alpha levels).

Then, using (12), membership values are obtained for the input variable in question, either for input (*e1, e2, de2* and *de1*) or output (*Ax1* and *Ax2*). For example, for a value $e1 = 115$ which represents a rigid value [variable x in (12)], the membership function for PS set, whose parameters have the values $a = 0$, $b = 63$ and c = 127, is:

$$triangle(x;a,b,c) = \max\left(\min\left(\frac{115-0}{63-0}, \frac{127-115}{127-63}\right), 0\right) = 0.1875$$

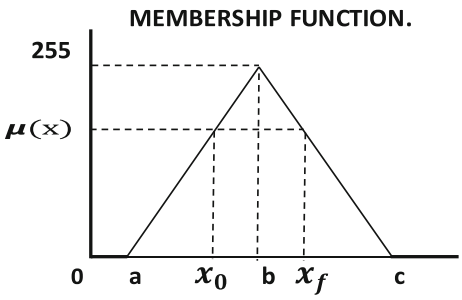

Fig. 7 Membership function using alpha-levels method for fuzzification

Table 1 Weight vectors matrix for input and output e_1 and de_1

Inputs	*Vectors*	*mf_1*	*mf_2*	*mf_3*	*mf_4*	*mf_5*
Data e_1	*Wnx=*	*Wnx_1*	*Wnx_2*	*Wnx_3*	*Wnx_4*	*Wnx_5*
Data de_1	*Wny=*	*Wny_1*	*Wny_2*	*Wny_3*	*Wny_4*	*Wny_5*

So, $\mu(x) = 0.1875$ is the membership value for that rigid input value x, in the fuzzy set PS. This is a floating point value that involves the use of more resources for its implementation in hardware. Therefore, a way to convert it to an integer value is using the following equation:

$$\left[\mu(x) = Eq.(12)_{floating\ point} \times 2^n - 1\right] = \mu(x)_{\text{integer value}}, \quad (13)$$

where $2^n - 1$ is the resolution (8 bits) for the system proposed. On substituting the membership value obtained and using (13), the membership value is:

$$\mu(x)_{escaled} = 0.1875 \times 127 = 23.8 \cong 24$$

This manner, the membership value scaled for $e1 = 115$ is $\mu(x) = 24$. This procedure is followed for each universe of discourse for the inputs and outputs. The values obtained are stored in look-up tables in the FPGA through internal program. Thus, the design application running does not have to calculate the membership value on chip. As an example, the VHDL code to obtain all membership values for NB set is:

type Negative_Big is array (a to c) of integer range 0 to $2^n - 1$.
constan f_m1:Negative_Big: = (a =>0,1 =>6,2 =>12,4 =>18,….,c =>$2^n - 1$);
for i in a to c loop
if (x = i) then
μ(x) = f_m1(i)
end if;
end loop;

where a and c are the parameters of the membership function given by (12).

2.3 Rules

In a fuzzy logic controller a descriptive verbal rules *If–Then* are used to describe the relation among inputs and outputs, according to:

$$R^{(l)} : IF\ x_1\ is\ F_1^l\ and...and\ x_n\ is\ F_n^l\ THEN\ y\ is\ G^l \quad (14)$$

Where

$(x_1 \ldots x_n)$ Represent the input variable
(y) Represent the output variable

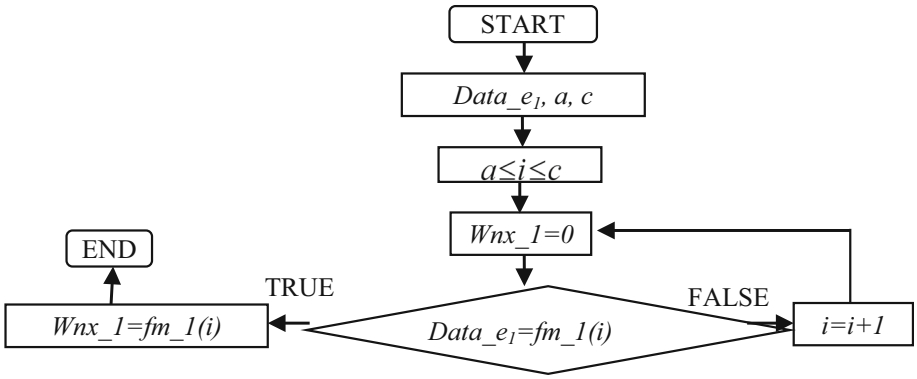

Fig. 8 Flow diagram for the fuzzification process

Table 2 Fuzzy logic rules for the FLC

e1	*de1*				
	NB	NS	Z	PS	PB
NB	NB	NS	NS	NS	NS
NS	NS	Z	PS	NS	Z
Z	NS	PS	Z	PB	PS
PS	Z	PSP	PS	Z	PS
PB	PS	PS	PS	PS	PB

(*FyG*) Represent the membership function of the fuzzy set

So, according to (14) and Mamdani method, 25 rules are generated. Each rule is obtained through the combination of each fuzzy set contained in the universe of discourse of the input (*e1*), with each fuzzy set contained in the universe of discourse of the input (*de1*), and then this combination is related to an output (*Z*), according to (14). Each rule is translated into a fuzzy relation which results in linguistic variables as shown in Table 2, which describe the relationships between rules for inputs *e1* and *de1*. The same method is used for inputs *e2* and *de2*.

2.4 Inference

Inference matrix is obtained by combining the weight vectors *Wnx* and *Wny* through the *min* operator according to Mamdani method [25–28]:

$$\mu_{A\to B}(x,y) = \min[\mu_A(x), \mu_B(y)] \tag{15}$$

It involves the comparison between two membership values reached in each rule and selecting the minimum value using (15). This is the first membership value of the input *e1* which is compared with each membership value of the input *de1* and so on. The process is done for all fuzzy set according to Table 1. Then, in order to program the inference process on the FPGA, a matrix is programmed which is used to store the minimum membership functions extracted according to (15). From now, this matrix will be referred as the membership values because it will be the result of the all rules with membership values different to zero or not, which corresponding to the membership function for the control action. Figure 9 shows the

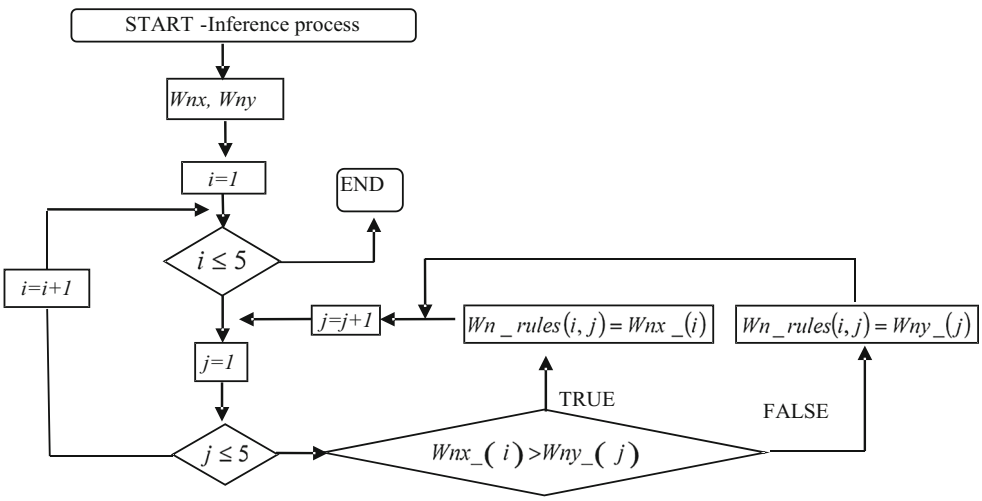

Fig. 9 Flowchart for the inference process

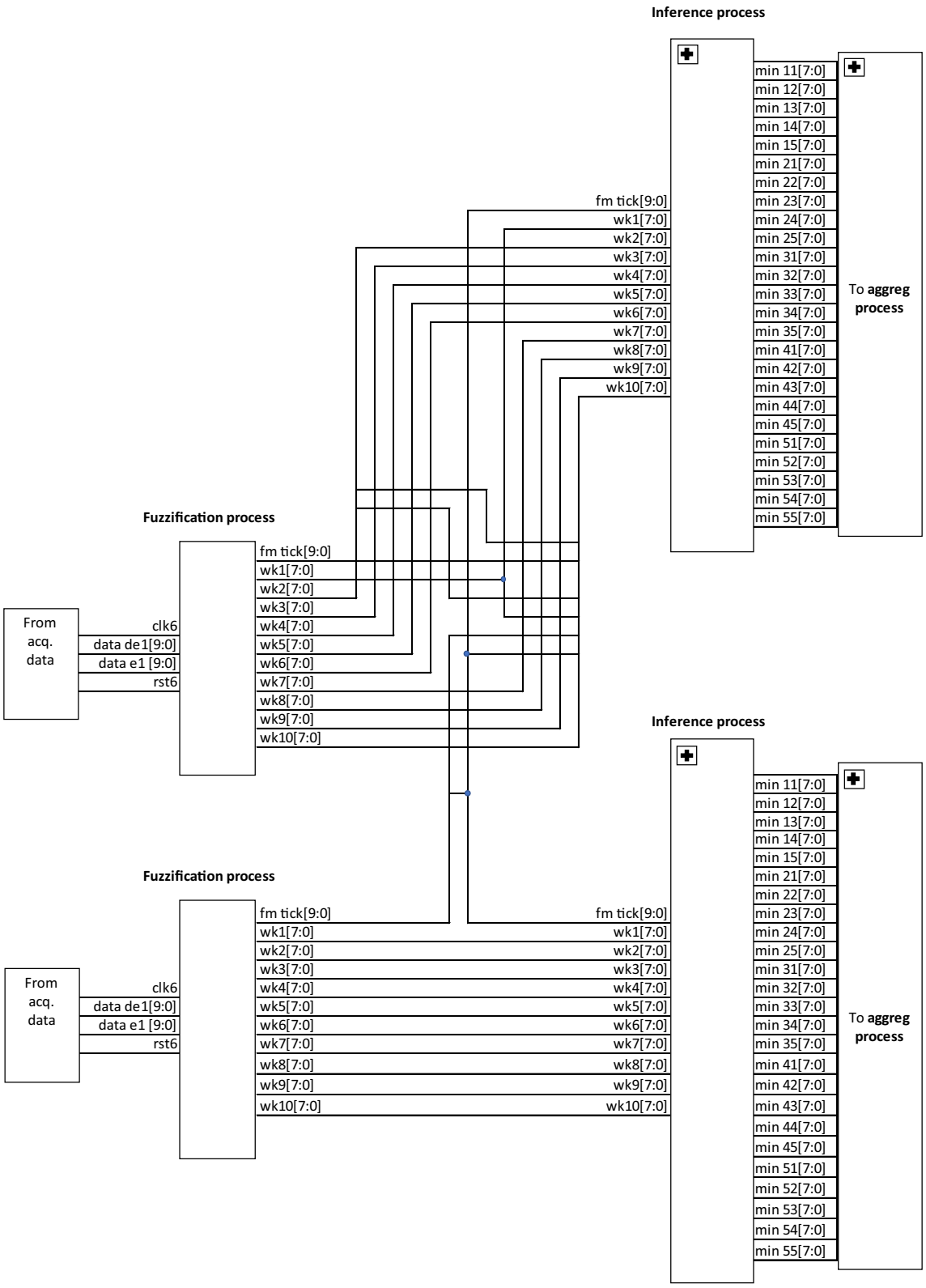

Fig. 10 Schematic blocks for the fuzzification and inference process

Table 3 Process sequence for aggregation

	e1				
	NG	NP	Z	PP	PG
De1					
NG	Wn_rules (1,1)	Wn_rules (1,2)	Wn_rules (1,3)	Wn_rules (1,4)	Wn_rules (1,5)
	↓	↓	↓	↓	↓
NP	Wn_rules (2,1)	Wn_rules (2,2)	Wn_rules (2,3)	Wn_rules (2,4)	Wn_rules (2,5)
	↓	↓	↓	↓	↓
Z	Wn_rules (3,1)	Wn_rules (3,2)	Wn_rules (3,3)	Wn_rules (3,4)	Wn_rules (3,5)
	↓	↓	↓	↓	↓
PP	Wn_rules (4,1)	Wn_rules (4,2)	Wn_rules (4,3)	Wn_rules (4,4)	Wn_rules (4,5)
	↓	↓	↓	↓	↓
PG	Wn_rules (5,1)	Wn_rules (5,2)	Wn_rules (5,3)	Wn_rules (5,4)	Wn_rules (5,5)

flowchart for the process. All membership values are stored in the matrix.

The schematic blocks for the fuzzification and inference processes implemented on the FPGA are shown in Fig. 10.

2.5 Aggregation

The aggregation stage is carried out by the method of maximum membership value for each rule in the inference stage whose value is different to 0, given by:

$$\mu_{B^l}(y) = \max\left[\mu_{F_1^l}(x_1), \ldots, \mu_{F_n^l}(x_n)\right] \tag{16}$$

It represents the union of the activated rules for each column contained in the matrix. In this case, five columns take into account only those rules that have a nonzero value. For this case, all values contained in a column are compared and the maximum value is selected as shown in Table 3. The arrows indicate the direction of the process to get the maximum. The process is performed for each column. The values obtained for each column represent the aggregation vector which is used for the defuzzification stage. This stage converts the fuzzy value to a real scalar value or rigid value. The flowchart for aggregation algorithm is shown in Fig. 11, where a vector called "*agreg*" stores the maximum membership values for each column. These membership values represent the output fuzzy sets.

2.6 Defuzzification

Defuzzification process is realized by the center of gravity according to [25–28]:

$$Z = \frac{\sum_{l=1}^{M} y^{-l}(\mu_{B'}(y))}{\sum_{l=1}^{M}(\mu_{B'}(y))}, \tag{17}$$

where y^{-1} represents the center of the aggregation vector and $\mu_{B'}(y)$ is the maximum membership value searched by the process.

This method requires $M = 2^n - 1$ iterations according to the input universe and is given by 2^n, where n is the number of bits used for the resolution [29]. Another method used in this work is the center average portions areas (COSAA) [24]. This method is suited for working with *alpha levels* and requires only $\alpha_{\max}$ iterations, given by:

$$z_{\text{COSAA}} = \frac{\sum_{i=0}^{\alpha_{\max}} \left(\frac{\left(x_f^{\alpha_f} - x_0^{\alpha_i}\right)}{2} + x_0^{\alpha_i} \right)}{\alpha_{\max}} \tag{18}$$

where $\alpha_{\max}$ is the highest membership value reached by the aggregation vector obtained in the aggregation stage.

This method takes into account the averages of the corresponding alpha levels and yours initial point value (alpha-level initial) and final point value (alpha-level final) for each output set contained in the aggregation vector. Then, using (18), the defuzzification values are obtained.

3 Experimental results

In order to verify the results, the FLC for two axes solar tracking on FPGA was tested through a scaled prototype which is shown in Fig. 12. The solar array is made up of three solar cells of 3.3 volts at 150 mA each. The module is mounted on a shaft profile base in which is anchored the servomotor for the rotation altitude axis and on this, is anchored the servomotor for solar azimuth rotation, as shown in Fig. 12. The sensors are mounted on a plastic base with a lid, which serves as a barrier between them. The bases are coupled through hydraulic pvc pipes in order to protect the cables from the bases to the box, where the FPGA Nexys2 is located, as shown in Fig. 13. This array allows to determine the error with more accuracy. Also, the Nexys2 board was shielded in a plastic enclosure 40 × 20 × 6 cm, with perforations for the cables and sensors connection.

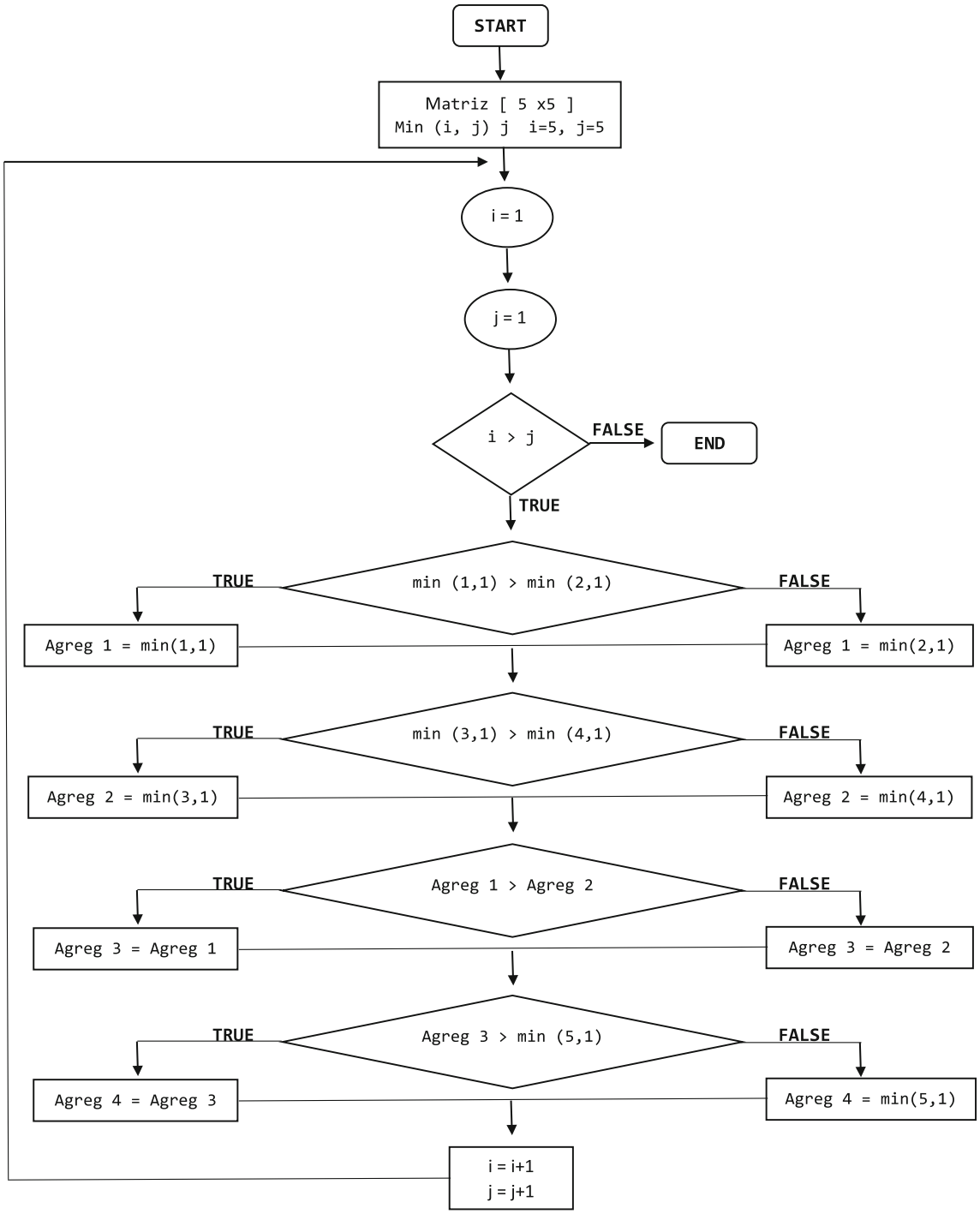

Fig. 11 Flow diagram for the aggregation process

The location where the tests were done is located in 19.33235925° north latitude and 99.18087544 longitude—east. Figures 14 and 15 show the graphs of solar angles for latitude and longitude for this location [23].

Experimental tests were made from 10:00 a.m. to 17:00 p.m. Furthermore, the resulting comparative graphs are in these intervals. First step was to test the synchronization in the system. Figure 16 shows the pulse *Cs* used to synchronize the data conversion from the light sensors to the FLC. As can be seen, there is a distortion which is due to noise environment or other external factor; however, it does not represent any problem to activate the conversion. Figure 17 shows the range of serial data sent by the sensors to the FPGA. Also, it has distortion in their construction;

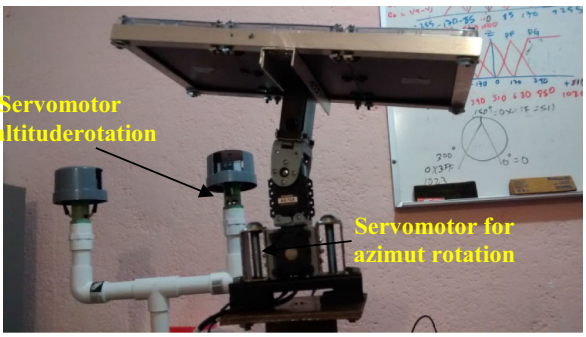

Fig. 12 Sun tracking prototype

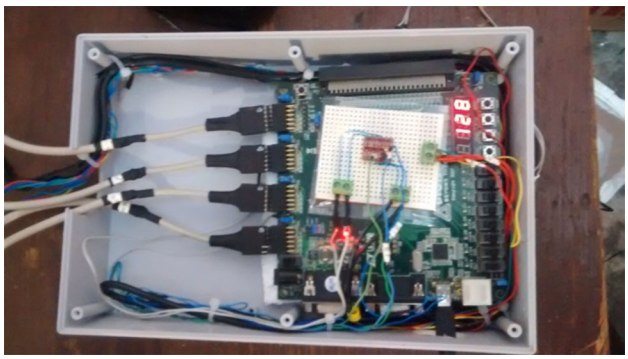

Fig. 13 FPGA Nexys2 in box

however, this does not affect the data acquisition stage. Figure 18 shows the data packet serially obtained at the output of the fuzzy controller, which is sent to the servomotor 1 for performing the control action. The result is similar to the second servomotor.

Figures 19 and 20 show the comparison between calculated and experimental results for azimuth and altitude angles obtained. The error is the deviations that exist between the two curves, both for those of Fig. 19 and for those of Fig. 20. The curve labeled "azimuth cal" was obtained theoretically using the equations of the solar angles, while the curve labeled "azimuth FLC" was obtained with the proposed FLC implemented in the solar panel system. Similarly, it is for Fig. 20, but for altitude angle. The most significant changes occur between 13:00 p.m. and 15:00 p.m. for both solar altitude and azimuth angle because in this period clouds were passing and block the light sensors which determine the input error to the fuzzy logic controller. As can be seen in both figures, the controller performs the control action with minimal errors, which for this application are quite good, since it is able to maintain, during the day, the solar panel always with the incidence of the rays of the sun at 90°, regardless of obstructions due to clouds or external factors. The experimental results were obtained with real time.

Simulated and experimental values measured at the output of the FLC for the sun tracking are shown in Table 4. These values are compared with simulated results using MATLAB toolbox. The test setup consisted of inputs *e1* and *de1* with values in the range of – 127–127 (first column in Table 4) and the defuzzified output with the method of mass center (column 2), alpha levels (column 3) and centroid. The mass center and centroid were obtained from the MATLAB toolbox. It should be mentioned that the FLC tests implemented in the FPGA were made using alpha levels. For each pair of input values (*e1, de1*), the results of each defuzzification method are shown in columns 2, 3 and 4. As can be seen, according to output values, the proposed design does not present a significant variation and does not affect the control action referred to the sun tracking, maintaining the solar incidence at 90° with respect to the plane of the PV system. The difference between them is less than 5% for all parameters over a full range contained in the universe of discussion. It can be concluded that the values obtained with the proposed FLC have a good approximation based on the theoretical and

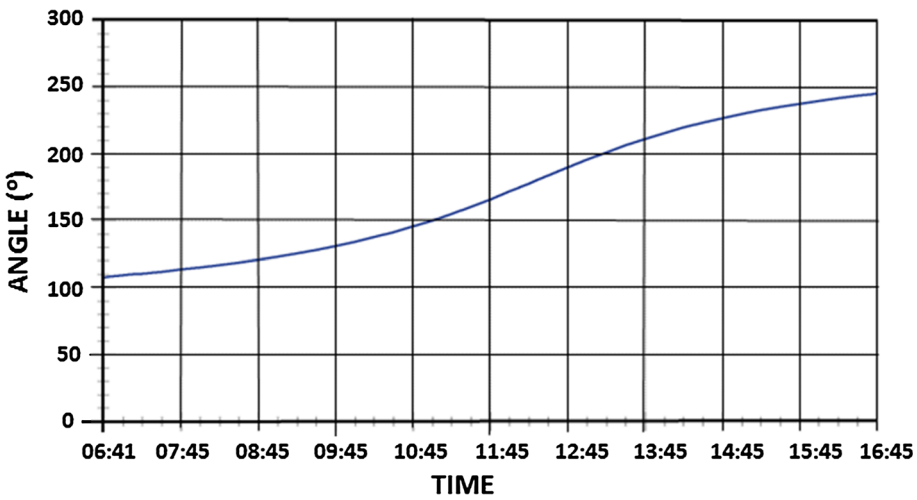

Fig. 14 Solar azimuth angles at November 9, 2014

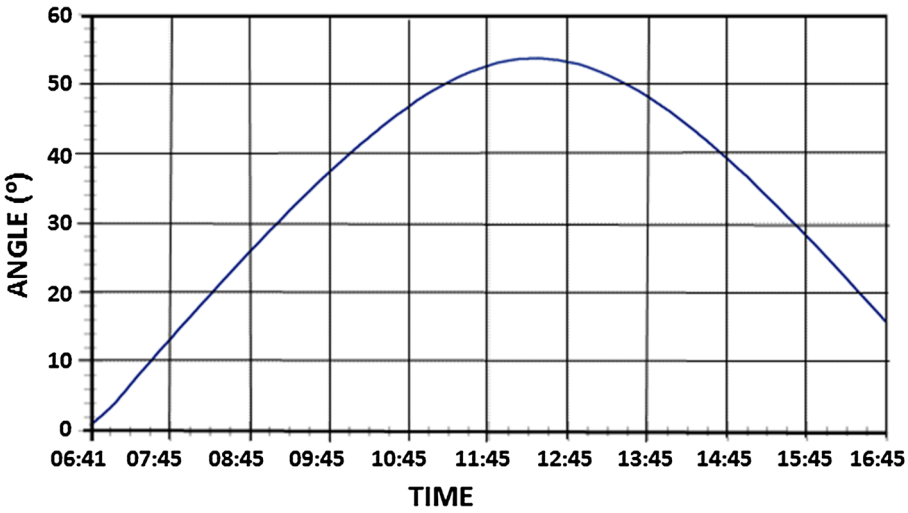

Fig. 15 Solar altitude angles at November 9, 2014

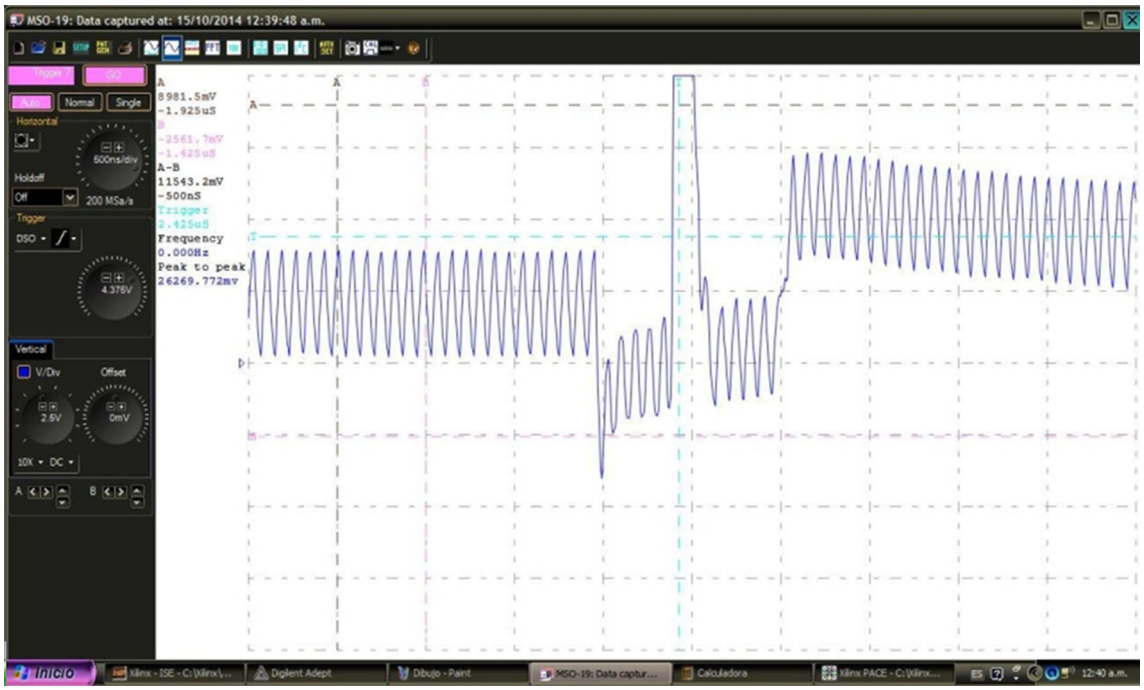

Fig. 16 Pulse *Cs* used to synchronize the data conversion from the sensors lighting to the FLC

simulated results. The stability is obtained when the controller reaches the convergence, that is, when the approximation error in the desired control action is minimal. The robustness of this system in terms of hardware is tested with the family of FPGA developed and designed by Diligent. The robustness in this case consists in performing the control action of the sun tracking on two axes, even with interfering signals in the system, because the FLC itself adapts it in each iteration.

4 Conclusions

This paper provides a new implementation of a FLC on FPGA for two axes sun tracking which resembles servomotors to control a PV system. The most important target of this work is a demonstration of a hardware design on FPGA for a sun tracking in order to increase the solar radiation received by a PV system. The fuzzy logic control is used to estimate the sun position at different conditions

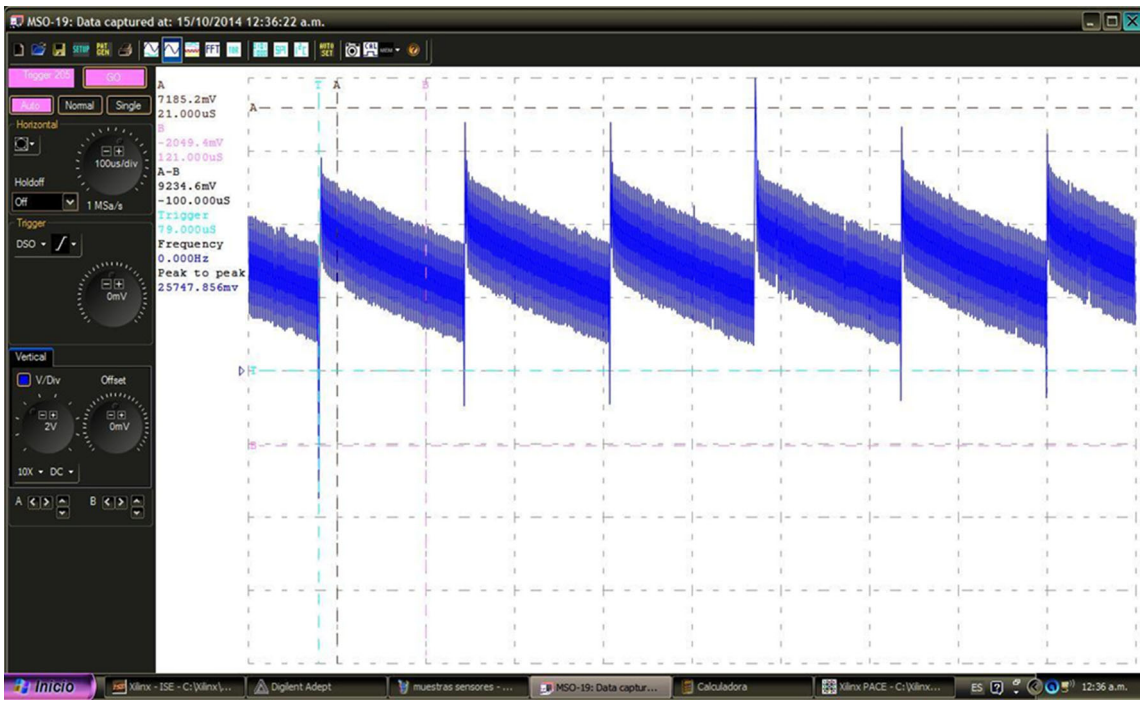

Fig. 17 Serial data sent by the sensors to the FPGA

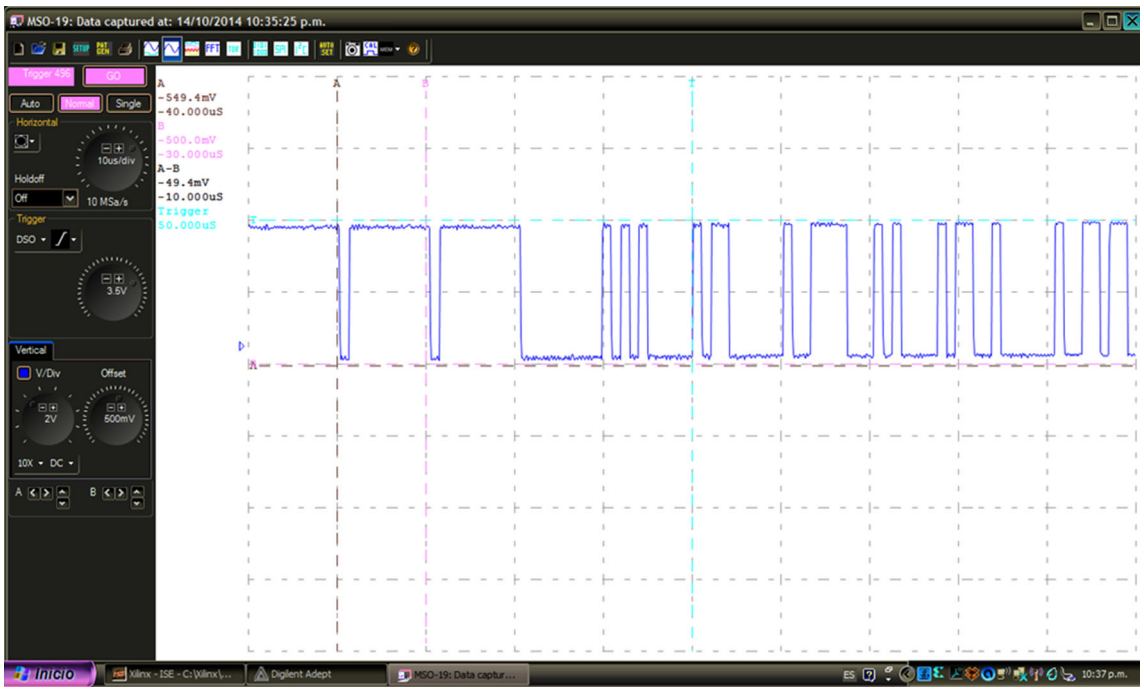

Fig. 18 Data packet serially obtained at the output of the FLC

in a day. The present approach proves that the fuzzy control design has a good performance and reduces the total number of operations, because look-up tables replace adders, multipliers and delay elements. Measurements at the input and output terminals indicate that there are distortions by external effects such as noise; however, these do not directly affect the performance of the controller. All simulation and experimental results demonstrate the

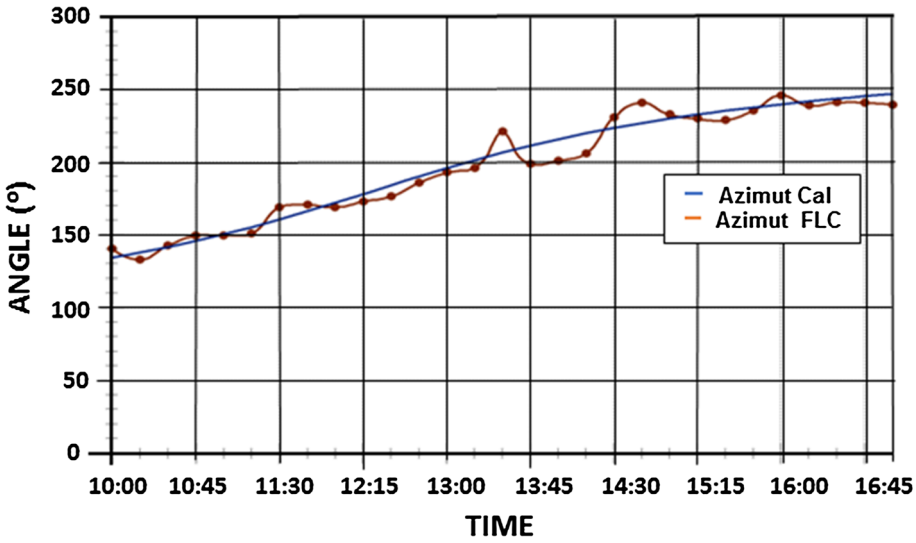

Fig. 19 Comparison between the azimuth angle calculated and obtained by the fuzzy logic controller

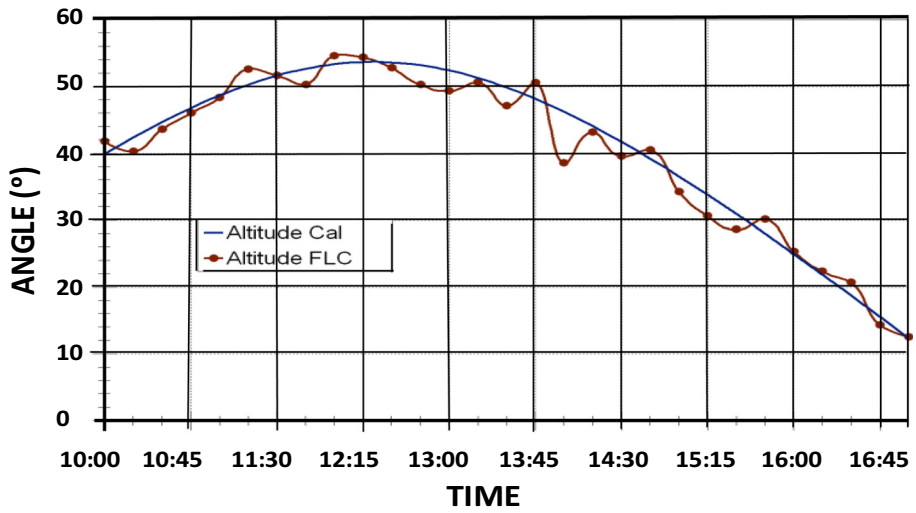

Fig. 20 Comparison between the altitude angle calculated and obtained by the fuzzy logic controller

Table 4 Comparison between different defuzzification methods

Inputs *e1*, *de1*		Mass centers	Alpha levels	Centroid MATLAB
− 127	− 127	0	0	81.9
− 100	− 100	141	155.5	213
− 92	− 85	197	170	238
− 35	− 25	411	384	391
− 15	− 10	472	384	484
25	25	613	640	618
35	40	665	640	661
50	75	796	843.5	768
60	90	843	858.5	768
70	95	862	863.5	770
85	100	883	868.5	788
110	100	883	868.5	830
127	115	954	883.5	934
127	127	1023	1023	942

effectiveness of the FLC in the PV system. Moreover, it was observed that the resolution with the servomotors used is highest in terms of requirements involving solar tracking. Also, because the movement of the sun is slow, the processing speed of the FPGA contributes to a better response of the system control. It can be concluded that the values obtained with the proposed FLC have a good approximation based on the theoretical and simulated results. So, the FLC can be used to estimate the sun tracking of a wide variety of PV systems. This approach leads to an efficient sun tracking.

Compliance with ethical standards

Conflict of interest The authors state that they have no conflict of interest with the publication of this research paper. Likewise, they also would like to thank the reviewers for their valuable comments and suggestions to improve the present work.

References

1. Youssef A, El-Telbany M, Zekry A (2017) The role of artificial intelligence in photo-voltaic systems design and control: a review. Renew Sustain Energy Rev 78:72–79
2. Antonio L, Steven H (2003) Handbook of photovoltaic science and engineering institute of energy conversion, University of Delaware, USA. Wiley Editorial, New York, pp 1020–1028
3. Sumathi V, Jayapragash R, Bakshi A, Akell PK (2017) Solar tracking methods to maximize PV system output: a review of the methods adopted in recent decade. Renew Sustain Energy Rev 74:130–138
4. Chavez UE, Yuri V (2013) Investigation of solar hybrid electric/thermal system with radiation concentrator and thermoelectric generator. Int J Photoenergy 2013:1–7
5. Njoku HO (2014) Solar photovoltaic potential in Nigeria. J Energy Eng 140:1–7
6. Zadey S, Dutt S (2013) Design of converter for low power photovoltaic conversion system. Int J Adv Res Electr Electron Instrum Eng 2(6):2733–2739
7. Aldali Y, Celik AN, Muneer T (2013) Modeling and experimental verification of solar radiation on a sloped surface, photovoltaic cell temperature, and photovoltaic efficiency. J Energy Eng 139:8–11
8. Abu-Rub H, Iqbal A, Ahmed Sk M (2012) Adaptive neuro-fuzzy inference system-based maximum power point tracking of solar PV modules for fast varying solar radiations. Int J Sustain Energy 31(6):383–398
9. Chekired F, Larbes C, Mellit A (2014) Comparative study between two intelligent MPPT-controllers implemented on FPGA: application for photovoltaic systems. Int J Sustain Energy 33(3):483–499
10. Gad HH, Haikal AY, Ali HA (2017) New design of the PV panel control system using FPGA-based MPSoC. Sol Energy 146:243–256
11. Wong J, Lim Y, Morris E (2015) Novel fuzzy controlled energy storage for low-voltage distribution networks with photovoltaic systems under highly cloudy conditions. J Energy Eng 141(1):1–15
12. Ansari MF, Chatterji S, Iqbal A (2010) A fuzzy logic control scheme for a solar photovoltaic system for a maximum power point tracker. Int J Sustain Energy 29(4):245–255
13. Altas IH, Sharaf AM (2008) A novel maximum power fuzzy logic controller for photovoltaic solar energy systems. Renew Energy 33(3):388–399
14. Barsoum N (2009) Implementation of a prototype for a traditional solar tracking system. In: Computer modeling and simulation, 2009. EMS'09. Third UK sim european symposium, IEEE, pp 23–30
15. Batayneh W, Owais A, Nairoukh M (2013) An intelligent fuzzy based tracking controller for a dual-axis solar PV system. Autom Constr 29:100–106
16. Chekired F, Larbes C, Rekioua D, Haddad F (2011) Implementation of a MPPT fuzzy controller for photovoltaic systems on FPGA circuit. Energy Procedia 6:541–549
17. Ahmad S, Shafie S, Ab Kadir MZA (2013) Power feasibility of a low power consumption solar tracker. Procedia Environ Sci 17:494–502
18. Prasanth Ram J, Rajasekar N, Miyatake M (2017) Design and overview of maximum power point tracking techniques in wind and solar photovoltaic systems: a review. Renew Sustain Energy Rev 73:1138–1159
19. Othman AM, El-arini MM, Ghitas A, Fathy A (2012) Real world maximum power point tracking simulation of PV system based on fuzzy logic control. NRIAG J Astron Geophys 1(2):186–194
20. Patyra MJ, Mlynek DM (2012) Fuzzy logic: implementation and applications. Wiley Editorial, New York, pp 237–242
21. Ozcelik S, Prakash H, Challoo R (2011) Two-axis solar tracker analysis and control for maximum power generation. Procedia Comput Sci 6:457–462
22. Tsai HL, Tu CS, Su YJ (2008) Development of generalized photovoltaic model using MATLAB/SIMULINK. In: Proceedings of the world congress on engineering and computer science WCECS, pp 1–6
23. Mendieta Garrido A (2013) Design of a PV plant for the Institute of Technology and Higher Education of Ecatepec, Master Dissertation
24. Hernández Zavala MA (2009) Arquitectura de Alto Rendimiento para Procesadores Difusos. Doctoral Dissertation, IPN, Mexico
25. Lughofer E, Sayed-Mouchaweh M (2015) Autonomous data stream clustering implementing split-and-merge concepts: towards a plug-and-play approach. Inf Sci 304:54–79
26. Lughofer E (2012) Hybrid active learning for reducing the annotation effort of operators in classification systems. Pattern Recognit 45:884–896
27. de Jesus Rubio J, Ochoa G, Meda JA, Rangel VI, Pacheco J (2015) Acquisition system and analytic fuzzy model of a manufactured wind turbine. IEEE Lat Am Trans 13(12):3879–3884
28. Rubio JJ, Bouchachia A (2016) MSAFIS: an evolving fuzzy inference system. Soft Comput. doi:10.1007/s00500-015-1946-4
29. Yurkovich S, Widjaja M (1996) Fuzzy controller synthesis for an inverted pendulum system. Control Eng Pract 4(4):455–469