MAXIMUM POWER POINT TRACKING USING MODIFIED P&O METHOD FOR THE OFF GRID RADAR

by

Carlos Andrés Giraldo Castañeda

A thesis submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE
in
ELECTRICAL ENGINEERING

UNIVERSITY OF PUERTO RICO
MAYAGÜEZ CAMPUS
2008

Approved by:

Sandra L. Cruz Pol, PhD Member, Graduate Committee	Date
José Colom Ustáriz, PhD Member, Graduate Committee	Date
Lionel Orama Exclusa PhD President, Graduate Committee	Date
Isidoro Couvertier Reyes, PhD Chairperson of the Department	Date
Francisco Maldonado, PhD Representative of Graduate Studies	Date

ABSTRACT

This work presents the design and implementation of Maximum Power Point Tracking (MPPT) techniques for photovoltaic arrays using a DSP embedded system for the application to Off-the-Grid Radars, providing a cost-effective solution which improves upon some of the constraints of previous methods. Two methods were implemented using modifications to Perturbation and Observation methods. The methods implemented are the Improved Perturbation and Observation (IP&O) and the proposed Selective Hopping (SH) method. The performance of each system was measured with respect to time convergence, power available, dynamic of the system and response to rapid variations of solar radiation. The techniques were evaluated in similar conditions in terms of temperature and solar radiation for a comparison. Both techniques used a hysteresis system that allows a decrease in the switching loss in the rapid variations of solar radiation.

The Selective Hopping technique allowed for improvement on some constraints of the IP&O technique. SH has a convergence time at least ten times faster than IP&O and it has approximately 5% more available power. The main problem with the SH method is that a local maximum power point emerges when the solar radiation changes instantaneously, for example (0-700 W/m2). However, these variations are not likely to happen in normal operation.

RESUMEN

Este trabajo presenta el diseño e implementación de técnicas de rastreo de máxima potencia de arreglos fotovoltaicos usando un sistema embebido con DSP para aplicación en los radares Off-the-Grid, proveyendo una solución costo efectiva la cual mejore algunas de las restricciones de métodos anteriores. Dos técnicas fueron implementadas usando modificaciones del método de Perturbación y Observación. Los métodos implementados son el método Mejorado de Perturbación y Observación (IP&O) y el método propuesto en esta disertación conocido como Método de Salto Selectivo (SH). El funcionamiento de cada sistema fue medido con respecto al tiempo de convergencia, potencia disponible, dinámica del sistema y respuesta ante variaciones rápidas de radiación solar. Las técnicas fueron también evaluadas bajo condiciones similares de radiación solar y temperatura para un estudio comparativo entre ellas. Ambas técnicas usan un sistemas de histéresis que permite disminuir las perdidas de conmutación en las variaciones rápidas de radiación solar.

La técnica de Salto Selectivo permitió mejorar algunas restricciones de la técnica IP&O El tiempo de convergencia de la técnica SH es por lo meno diez veces mas rápido que IP&O y obtuvo aproximadamente 5% mas de la potencia disponible. El principal problema del método de SH es un máximo local de potencia cuando ocurre un cambio instantáneo de la radiación solar, por ejemplo de 0-700w/m2. Sin embargo estas variaciones no son esperadas bajo la operación normal.

.

To God. . .

ACKNOWLEDGEMENTS

During the development of my graduate studies at the University of Puerto Rico several persons and institutions collaborated directly and indirectly with my research. Without their support I would have not finish this work.

I want to begin by expressing a sincere acknowledgement to my advisor, Dr. Lionel Orama Exclusa because he gave me the opportunity to conduct this research under his guidance and supervision. I received motivation; encouragement and support form him during all my studies. I want to thank to Dra. Sandra Cruz Pol and Dr. José Colom Ustáriz for teaching me excellent courses that contributed to my professional formation. Also I want to thank to CLIMMATE team for providing a friendly work environment.

This work was supported primarily by the Engineering Research Centers Program of the National Science Foundation under NSF award number 0313747. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author and do not necessarily reflect those of the National Science Foundation.

Last, but most importantly I would like to thank God "The giver of all the things", my family and my girlfriend, for their unconditional support, inspiration and love.

Table of Contents

Table List

Tables **Page**

Figure List

Figures **Page**

1 INTRODUCTION

The National Science Foundation Engineering Research Center known as the Collaborative Adaptive Sensing of the Atmosphere (ERC-CASA) [McLaughlin, 2003] is developing networks of radars for weather monitoring and prediction. The concept is called the Distributed Collaborative Adaptive Sensing (DCAS) [Victor Marrero, et al., 2005], [Jorge M. Trabal, et al., 2006] and [Manuel A. Vega, et al., 2006]. The DCAS is the deployment of small low-range power radars that communicate and adapt to collaborate during weather phenomena. The DCAS concept will probably replace the NextRad technology which uses just one large, high-power device that covers hundreds of miles. Since we will have many radars covering a relatively small area, good data resolution will be achieved. The first test bed is already in place and CASA researchers are starting to gather data for evaluation with the NexRad.

The concept of an Off-the-Grid Radar (OTG) has been proposed as part of a test bed under development by UPRM and UMASS students. The OTG is an important part of the DCAS. The OTG is a radar node that is tied to neither the power grid nor the communications network. Communications will be wireless and the power of the entire node will be generated on site using renewable energy sources such as a photovoltaic system. One big advantage of the OTG is that it will be deploy to places where the end users, like the National Weather Service, will need it the most to cover phenomena under study.

Recently, renewable energy systems have increased in importance for a variety of reasons. Among others, we need clean, natural and theoretically endless sources of power to move the world and its continuous changes. Renewable sources are reliable and help avoid health and environmental problems due to the close to zero pollution they generate. Another important reason is the need for energy in remote places, where the traditional power grid is not available or is not easy to deploy. The latter is why the OTG needs to be powered by a renewable energy alternative.

Photovoltaic systems (PV) are energy renewable sources and many different researchers have been working to maximize their power conversion, allowing the overall efficiency to be increased. Therefore, the PV system is an essential component in the design of Off-the-Grid radars.

Different techniques for the maximum power point tracking (MPPT) exist. A novel technique that uses modified Perturbation and Observation method [N. S. D'Souza, et al., 2005] [M.-L. Chiang, et al., 2002] has been considered and compared to previous designs reported in the literature.

1.1 Motivation

The energy produced by a photovoltaic system depends mainly on the solar radiation available on site as well as temperature variations. A variety of methods have been developed to enhance maximum power output of photovoltaic (PV) modules. These methods are known as Maximum Power Point Tracking methods (MPPTs). Many of these use mathematic models to make approximated descriptions of the PV modules behavior, and design the algorithms based on the models [Minwon Park, et al., 2004], [Bernard Bekker, et al., 2004] and [Nobuyoshi Mouth, et al., 2006]. Other methods use the standard test of a PV module to design the tracking systems. For example, the most common and simple method for MPPT is the Perturbation and Observation algorithm (P&O) [Joe-Air Jang, et al., 2005]. Other methods used the relationship between the short circuit current and the maximum power current [K. Samangkool, et al., 2005]. Most of those methods were tested with simulations and results approximate the real performance. In addition, the implementation of some methods is very complex, making the process computationally expensive.

The Maximum Power Point Tracking method proposed in this work will be reliable with a high versatility and low cost and overcomes some constraints in previous systems such as the convergence time and switching loss. In addition, the system will be a validation instrument for solar radiation on site.

The proposed Maximum Power Point Tracking is expected to be integrated with DC/DC converter at all nodes for Off-the-Grid radars, currently under development for the Collaborative Adaptive Sensing of the Atmosphere (CASA) [McLaughlin, 2003], a National Science Foundation Engineering Research Center (ERC).

1.2 Literature Review

Many different techniques for the MPPT of photovoltaic arrays have been developed. The Perturbation and Observation Method (P&O) [O. Wasynczuk, 1983], [Y.-T. Hsiao, et al., 2002] and [T. Tafticht, et al., 2004] is the most simple. Figure 1.1 shows a typical power curve of a photovoltaic array and the maximum power point.

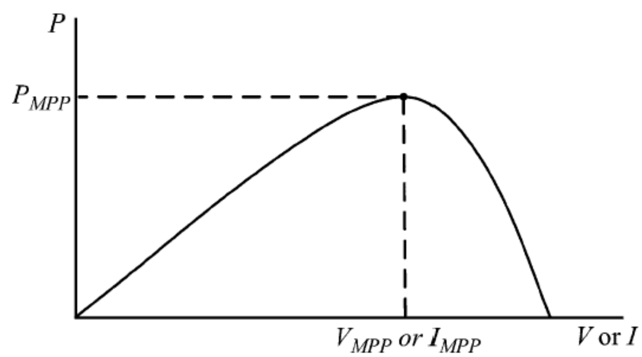


Figure 1.1 Characteristic PV array power curve [Joe-Air Jang, et al., 2005].

A flowchart of the P&O method is illustrated in Figure 1.2, which shows how the operating point moves toward the maximum power point, periodically increasing or decreasing the photovoltaic array voltage by comparing power quantities between the present state and the past history [Joe-Air Jang, et al., 2005], [Femia, et al., 2005], [Chee Wei Tan, et al., 2006] and [Femia, et al., 2004].

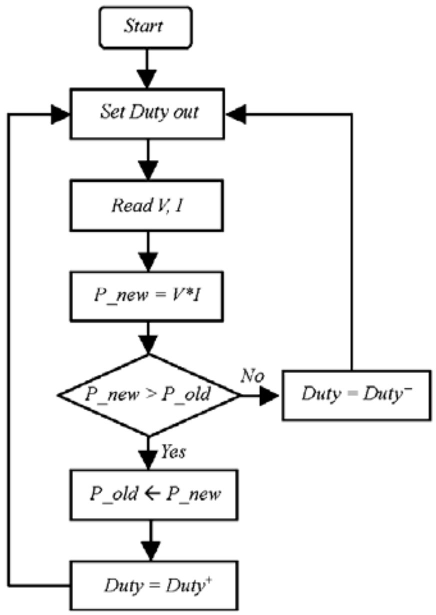


Figure 1.2 Flowchart of the P&O algorithm [Chee Wei Tan, et al., 2006].

The main problem of this method can be seen when atmospheric conditions (mainly solar radiation) change rapidly. These changes create an operating point of the system that appears far from the maximum power point, creating numerical instability, a possible convergence problem.

The IP&O method is an improved perturbation and observation method based on the adaptive algorithm proposed by [Youngseok Jung, et al., 2005] and [Ying-Tung Hsiao, et al., 2002]. IP&O automatically adjusts the reference step size and hysterics bandwidth for power comparison [Joe-Air Jang, et al., 2005], [Femia, et al., 2005], [Chee Wei Tan, et al., 2006] and [Femia, et al., 2004].

The flowchart of the IP&O method is illustrated in Figure 1.3, which shows the same tracking structure that its predecessor P&O possessed, with the addition of a hysteresis system that allows the deflections of the maximum power point to be reduced. A more detailed description can be found in [T. Senju, et al., 1994].

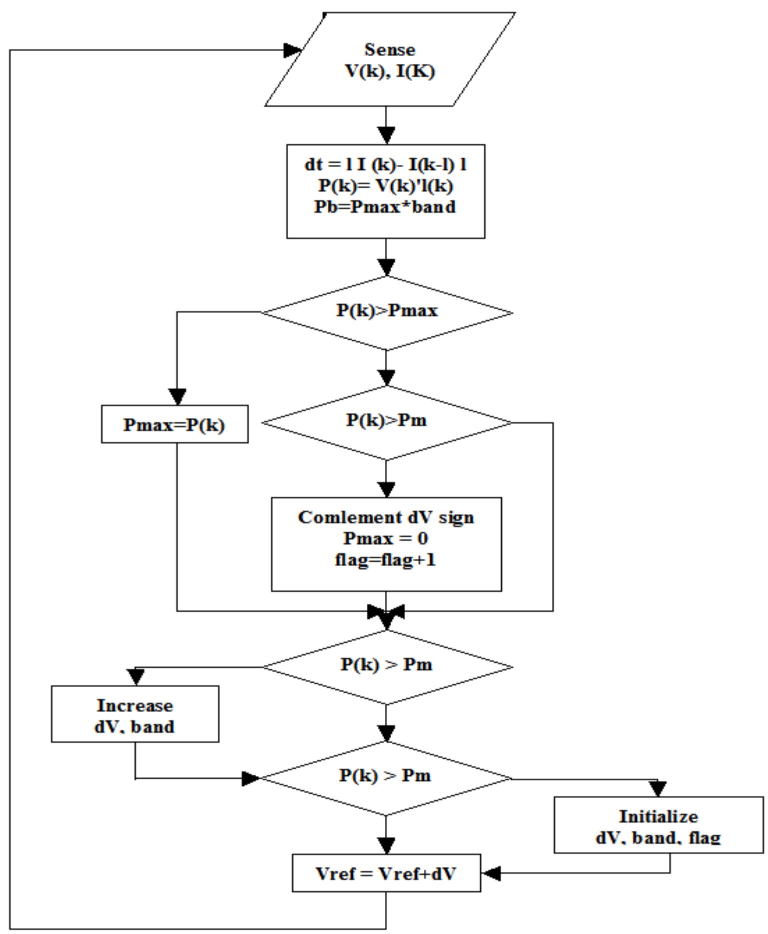


Figure 1.3 Flowchart of the IP&O method [Youngseok Jung, et al., 2005].

The main advantage of the IP&O method is that the total photovoltaic power output increases by 0.5% at an unsettled weather condition compared to the traditional Perturbation and Observation Method (P&O) [Joe-Air Jang, et al., 2005], [Chee Wei Tan, et al., 2006] and [Femia, et al., 2004]. A disadvantage of this method is the high computational cost. Additionally, implementation can be difficult and the convergence speed varies [Trishan Esram, et al., 2006].

The Fractional Short Circuit Current (FSCC) method [G. W. Hart, et al., 1984], [M. A. S. Masoum, et al., 2002] uses a relationship that exists between the photovoltaic panel's short circuit current (Isc) and the maximum power point current (I_{MPP}) [B. Bekker, et al., 2004] and [S. Yuvarajan, et al., 2003]. This relationship results from fact that, under varying atmospheric conditions, I_{MPP} is approximately linearly related to the Isc of the PV array as shown in [T. Noguchi, et al., 2000] and [N. Mutoh, et al., 2002]. The equation 1.1, which describes the method, follows:

$$I_{MPP} \approx \boldsymbol{K_{CURRENT}} * \boldsymbol{I_{SC}} \qquad \textbf{1.1}$$

Where $\boldsymbol{K_{CURRENT}}$ is proportionally constant and determined according to the PV array in use. The constant is generally found to be between 0.78 and 0.92. The main problem of FSCC method is measuring Isc during operation because the short current is periodically delivered to the power converter.

The Fractional Open Circuit Voltage (FOCV) method [J. J. Schoeman, et al., 1982], [M. Buresch, 1983], [D. J. Patterson, 1990] uses the approximately linear relationship between open-circuit voltage (V_{OC}) and maximum power point voltage (V_{MPP}) under varying conditions of temperature and solar radiation [H.-J. Noh, et al., 2002] and [K. Kobayashi, et al., 2004]. Equation 1.2, which describes the method, follows:

$$V_{MPP} \approx \boldsymbol{K_{VOLTAGE}} * \boldsymbol{V_{OC}} \tag{1.2}$$

The $\boldsymbol{K_{VOLTAGE}}$ usually has to be computed beforehand by empirically determining V_{MPP} and V_{OC} for the specific PV array at the different radiation and temperature levels. The factor has been reported to be between 0.71 and 0.78 [Trishan Esram, et al., 2006]. The main problem with the FOCV method is that V_{OC} is measured periodically, and in order to do this the power converter is momentarily shut down.

Neural Networks Methods have the potential to provide an improved method of deriving non-linear models which are complementary to conventional techniques [Hiyama, et al., 1995] and [L. Zhang, et al., 2002]. These networks have been used to control the photovoltaic panel non- linear behavior and its maximum power.

The structure of the proposed neural network used for MPPT is shown in figure 1.4. The network has three layers: input layer, hidden layer and output layer. The input layer has 3 neurons: one for array voltage, another for array current and one more for cell temperature.

The output layer has one neuron for the control boost converter (Up-converter DC/DC). The hidden layer has 300 neurons. The network is fully connected, i.e. the output of each neuron is connected to all neurons in the hidden layer through a weight which is not shown in the figure. Additionally, a bias signal is coupled to all the neurons through a weight [K. Samangkool, et al., 2005].

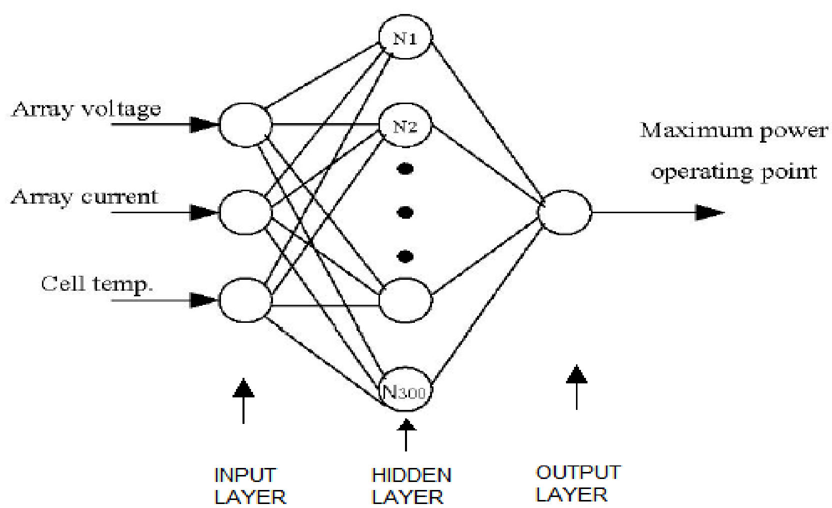


Figure 1.4 Neural network structure for MPPT weight [K. Samangkool, et al., 2005].

The neural networks are trained with solar radiation data and temperatures in the operating range. This is needed to track the performance of the system. The major problem of neural networks is the high complexity of the implementation [Trishan Esram, et al., 2006].

1.3 Proposed Method

The Selective Hopping (SH) method is based on P&O method and uses a improved tracking structure that allows selective steps in the power curve of the photovoltaic array, reducing the convergence time and keeping computational cost low. A hysteresis system was used, allowing a decrease in the deflections of the maximum power point. Chapter 2 provides a more detailed discussion of the SH method.

1.4 Organization of the Work

The study of several maximum power point tracking methods is presented in Chapter 1. Chapter 2 develops the design tests and presents the design of the Selective Hopping and Improved Perturbation and Observation methods. Chapter 3 shows the performance of each method (time convergence, dynamic, available power) and the behavior of the methods in similar conditions (temperature, solar radiation). In the fourth and final chapter, the conclusions and future works are presented.

2 MAXIMUM POWER POINT TRACKING DESIGN

This chapter presents the overall design of the maximum power point tracking system, consisting of hardware, software, and firmware. The hardware includes the design of the DSP embedded system, the load card, and the conditioner card. The firmware consists of two tracking algorithms (Improved Perturbation and Observation and the newly proposed Selective Hopping Method) and the hysteresis systems for both techniques. The software contains the graphical user interface.

2.1 Design Test

The main design criterion is the time stability of the photovoltaic array, which can be found using the circuit shown in Figure 2.1. The circuit is a switch system that allows the solar panel to short circuit for a short time and then open the circuit. This process is repetitive and allows the time required for the panel to deliver the maximum current to the load (Current Sense Resistor) to be measured.

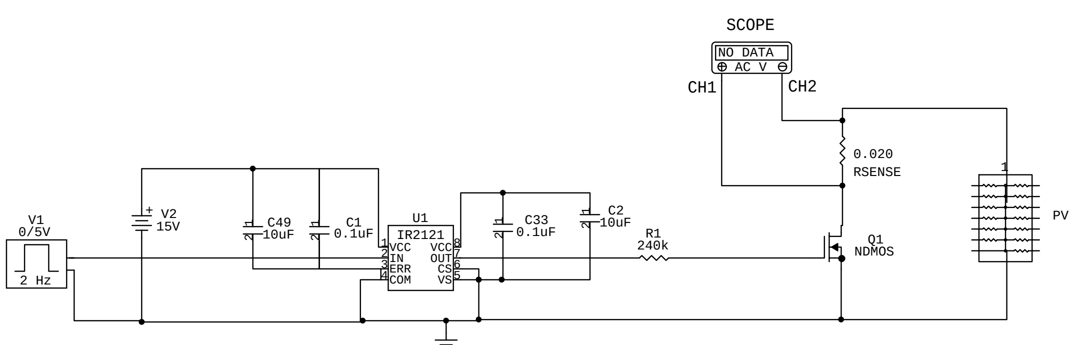


Figure 2.1 Test circuit for the time stability of PV.

The difference between on curve Channel 1 and Channel 2, which allows the stability time to be measured, was observed in the scope. Figure 2.2 shows the results of the test, finding an approximate stability time of 78 ms for the photovoltaic array BP 3160. This time is very important for the convergence of the tracking techniques. In the final design a time of (2 *stability time) ≈ 160 ms was used for the algorithm to find the point of maximum power.

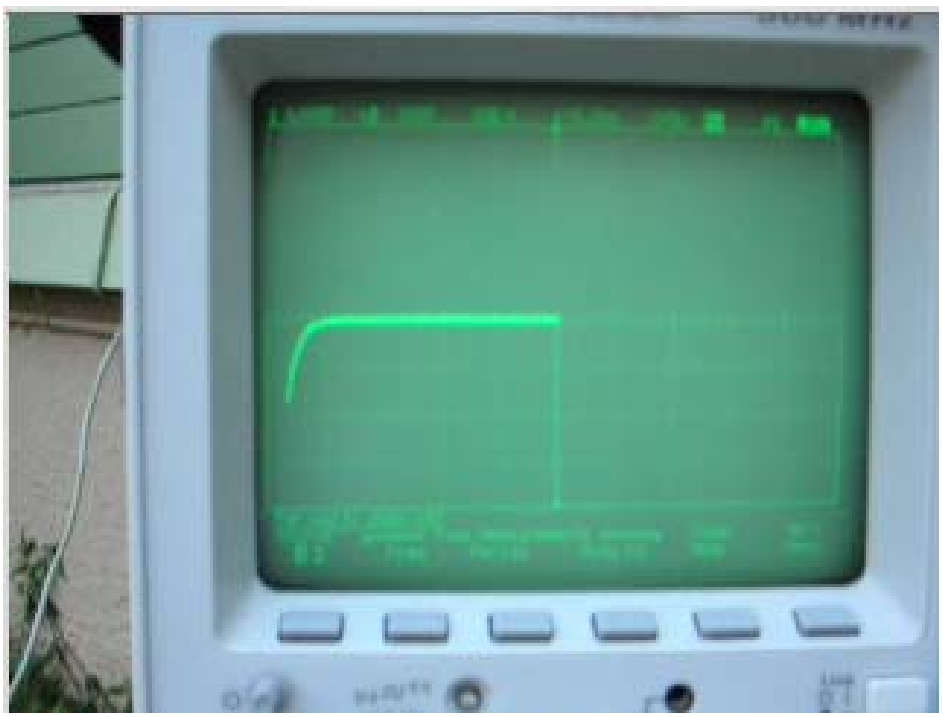


Figure 2.2 Stability time

2.2 Design of MPPT Measuring System

2.2.1 Overview

The track system consists in finding the point (voltage and current) of photovoltaic array that allows the maximum power to be delivered to the load. Therefore, we implemented a system that forces the PV to work in the optimal voltage and current characteristic. This is achieved with a resistive system that perturbs the photovoltaic array with controlled variations of the load, finding the matching load.

The technique for finding this point is the tracking method. Two techniques, the Selective Hopping and the Improvement Perturbation and Observation, are presented in this work. These techniques are implemented with the same hardware and software (user interface). The difference in the methods is the firmware that controls the embedded system.

The embedded system used a Digital Signal Processing Controller (dsPIC30F4013 of Microchip Technology) which is the overall controller of the system. It is used to provide the communication with the computer via the RS-232 interface, control the load, and acquire the voltage and current signals from the PV.

2.2.2 Hardware Design

This section presents the schematic diagrams for every part of the maximum power point tracking system. Figure 2.3 shows the embedded system, based on DSP, whose function is to provide the overall control of the system through the switching of the resistor set and acquire the voltage and current signals of the photovoltaic array for the A/D converters. The circuit in Figure 2.3 consists of the RS-232 interface, the DSPic30F4013 with all ports connected to the data bus, and the power supply. A MAX3221 device was used to configure the physical protocol for the RS-232 interface.

On the other hand, an RS-232 line driver is included to facilitate the communication between the embedded system and the PC. This connection operates at 115200 bps and employs a protocol to transfer the data correctly.

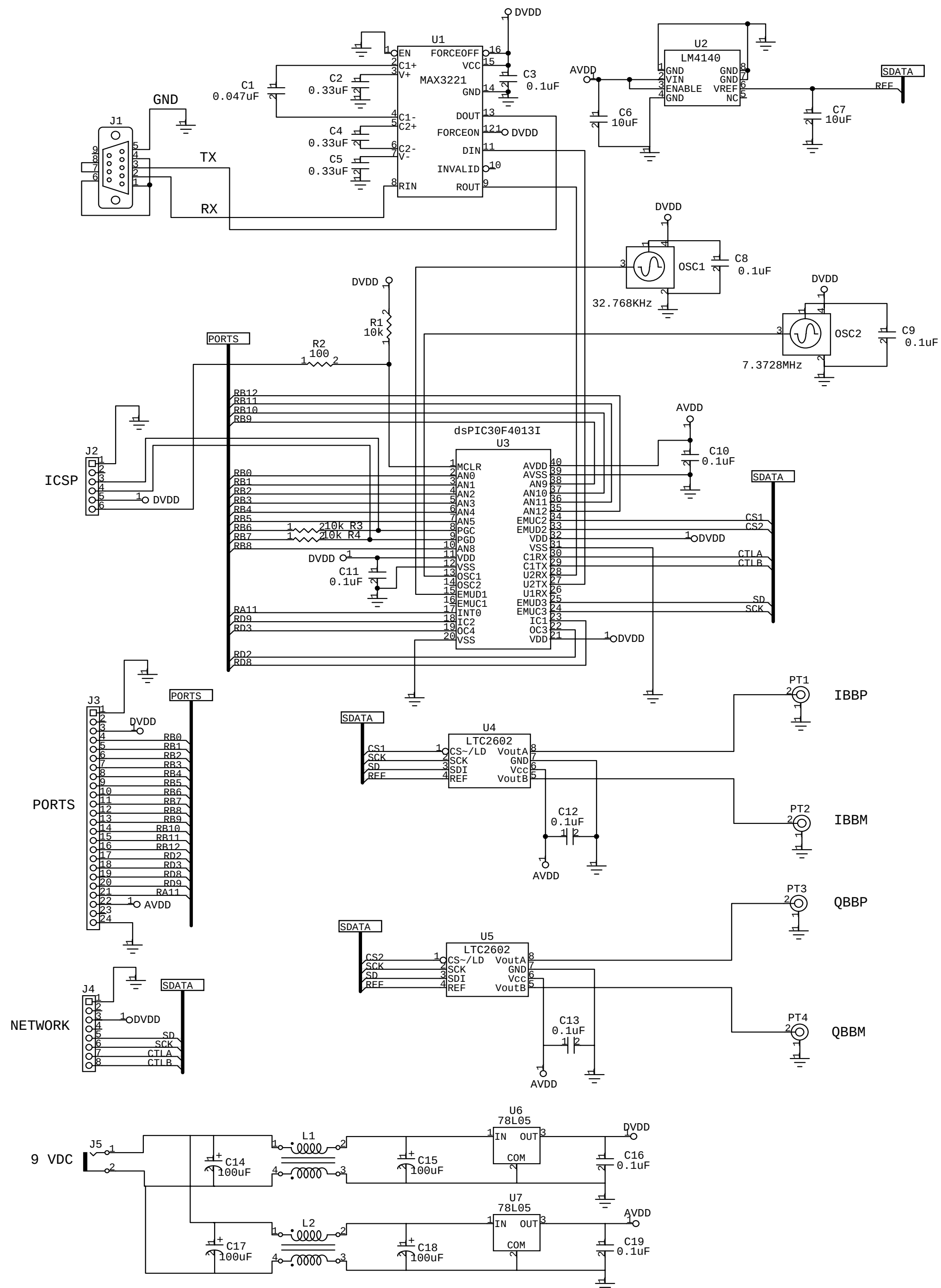


Figure 2.3 Embedded system based on DSP.

The load card is a resistor set digitally controlled by the embedded system, and allows the perturbation of the photovoltaic array like a tracking mechanism. The resistor set is a binary resistor system based on the binary structure where $20m\Omega$ is the base. Table 2.1 shows the structure, which can have 2^{15} resistor combinations. Column 3 shows the desired resistance with respect to the structure.

Table 2.1 **Binary resistor system.**

Resistance	Binary structure	Desired resistance (Ω)
R0	$2^0 * 20m\Omega$	$20*10^{-3}$
R1	$2^1 * 20m\Omega$	$40*10^{-3}$
R2	$2^2 * 20m\Omega$	
R3	$2^3 * 20m\Omega$	$160*10^{-3}$
R4	$2^4 * 20m\Omega$	$320*10^{-3}$
R5	$2^5 * 20m\Omega$	$640*10^{-3}$
R6	$2^6 * 20m\Omega$	1.28
R7	$2^7 * 20m\Omega$	2.56
R8	$2^8 * 20m\Omega$	5.12
R9	$2^9 * 20m\Omega$	10.24
R10	$2^{10} * 20m\Omega$	20.48
R11	$2^{11} * 20m\Omega$	40.96
R12	$2^{12} * 20m\Omega$	81.92
R13	$2^{13} * 20m\Omega$	163.84
R14	$2^{14} * 20m\Omega$	327.68
R15	$2^{15} * 20m\Omega$	655.36

Table 2.2 shows the commercial values of the resistor structure (see Column 4) with their corresponding powers (see Column 5). These values were found using the nominal electric characteristics of the photovoltaic array BP 3160.

Table 2.2 Commercial values of resistor structure.

Resistance	Ohms (Ω)	Power calculate (W)	Resistance commercials (Ω)	Power commercials (W)
R0	$20*10^{-3}$	1.28	0.020/0.020	1.5
R1	$40*10^{-3}$	2.56	0.040/0.040	3
R2	$80*10^{-3}$	5.12	0.080/0.040	3
R3	$160*10^{-3}$	10.24	0.16/0.15	35
R4	$320*10^{-3}$	20.48	0.32/0.33	35
R5	$640*10^{-3}$	40.96	0.64/0.33 0.30	35 35
R6	1.28	81.92	1.28/0.51 0.25	35 35
R7	2.56	163.84	2.56/0.51	35
R8	5.12	164.6	5.12/1.0 0.15	35 5
R9	10.24	82.13	10.24/3.3 0.15	35 3
R10	20.48	41.06	20.48/7.5 5.6	20 20
R11	40.96	20.53	40.96/15 10	20 20
R12	81.92	10.26	81.92/75 6.8	20 1
R13	163.84	5.133	163.84/150 12	20 0.5
R14	327.68	2.566	327.68/82	1
R15	655.36	1.283	655.36/150 56	0.5 0.5

Figure 2.4 shows the load card whose function is to allow the controlled load variations for the bus data. The card consists of multiple drivers (IR2121 of International Rectifier) and HEXFET transistors (IR3315 of International Rectifier) that form the switching system.

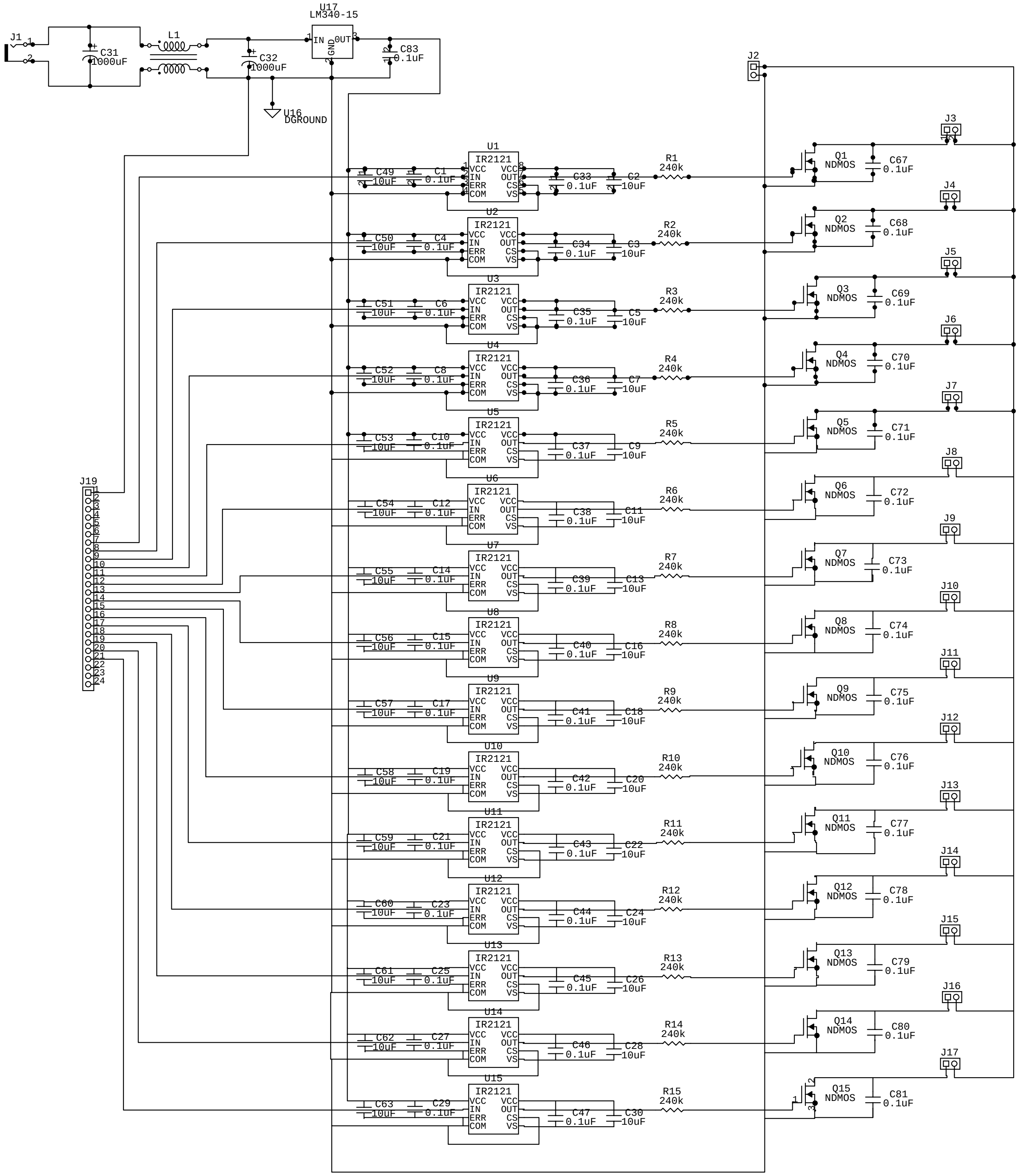


Figure 2.4 Load card.

A conditioner card provides sensing of the current and voltage of the photovoltaic array. This uses Current Sense Resistors and instrumentation amplifiers to condition the signals to the adequate levels for the A/D converters. Also this card provides a voltage reference of 2.5V to the DSP controller for the A/D converters. Figure 2.5 shows the conditioner card with all components.

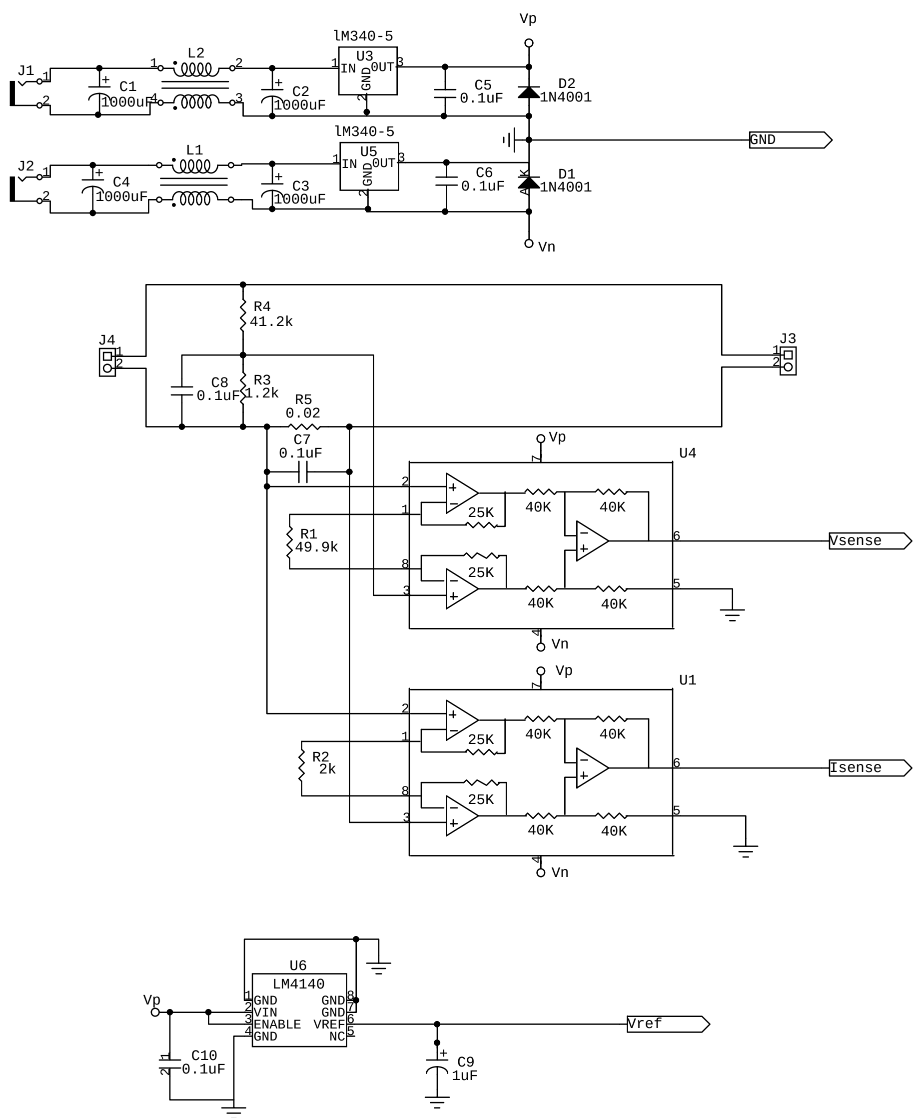


Figure 2.5 Conditioner Card.

Figure 2.6 shows a photograph of the whole maximum power point tracking system and components: the embedded system based on DSP, the conditioner card and the load card.

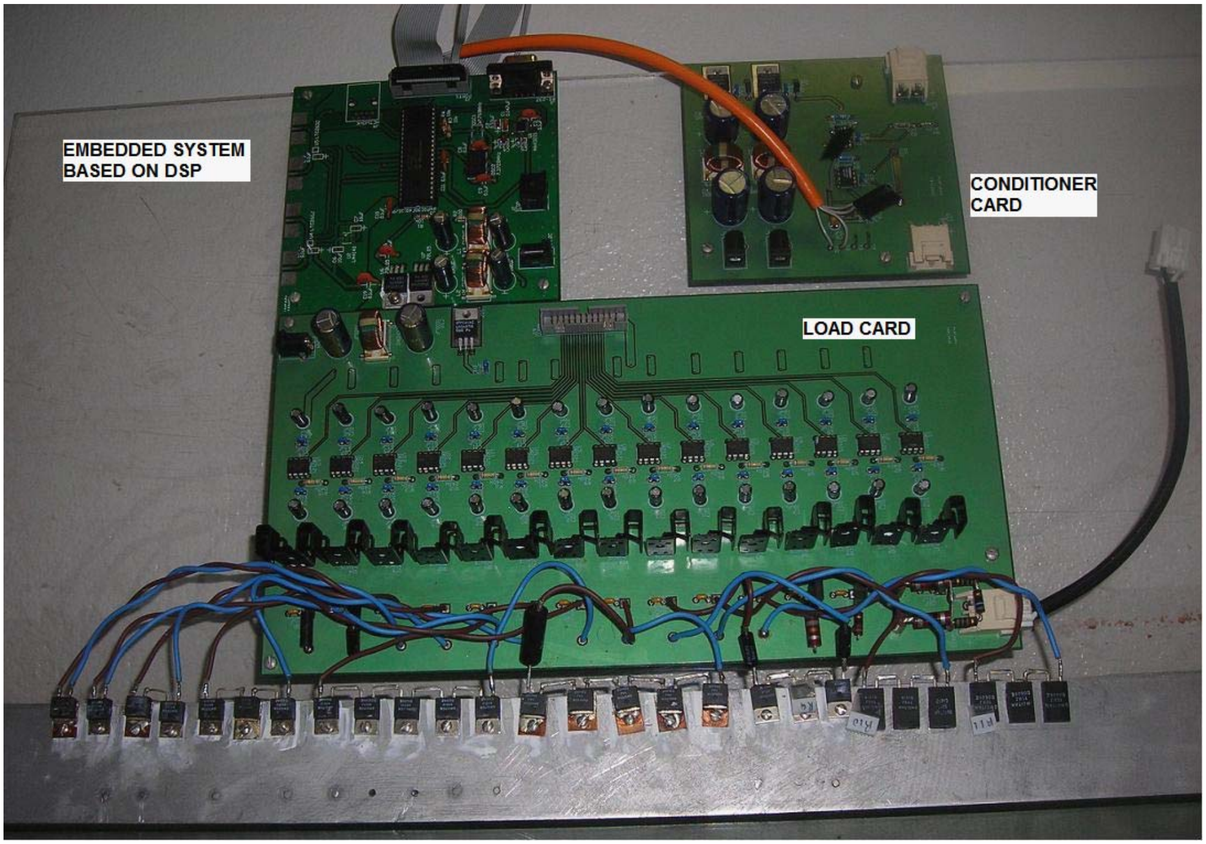


Figure 2.6 Maximum power point tracking components.

2.2.3 Hardware Operation

The user can control the tracking system in the graphic user interface by clicking on the "start" button. When this button is pushed, a control word is transmitted from the PC to the embedded system that is carried out through the RS-232 interface, employing a baud rate of 115200bps, 8-bit format and no parity. The data protocol is as follows:

First, an initialization byte (0xAA) is transmitted. The system begins to track the maximum power point and acquires the voltage and current from the A/D channels. The A/D conversion is 12 bits, where the first byte is the low part and the second byte is the high part of the A/D conversion. The data is stored in a double buffer structure with a size of 2*32 bytes for each buffer. This format allows the transmission from the embedded system to the PC. Figure 2.7 shows the structure the double buffer and the data storage format.

The embedded system generates one perturbation to the photovoltaic array every 160 ms, storing the voltage and current conversion in the buffer. When the first buffer is full, it generates an interruption, and transmits all data on the buffer to the PC. It simultaneously stores the new data in the second buffer. If the second buffer is full, the embedded system transmits the contents of the buffer to the PC and stores the new data in the first buffer. This process is cyclic and buffer transmission time is 4.44 ms, which prevents data loss. Table 2.3 provides more detailed information about A/D conversion time, transmission time, buffer transmission time, etc.

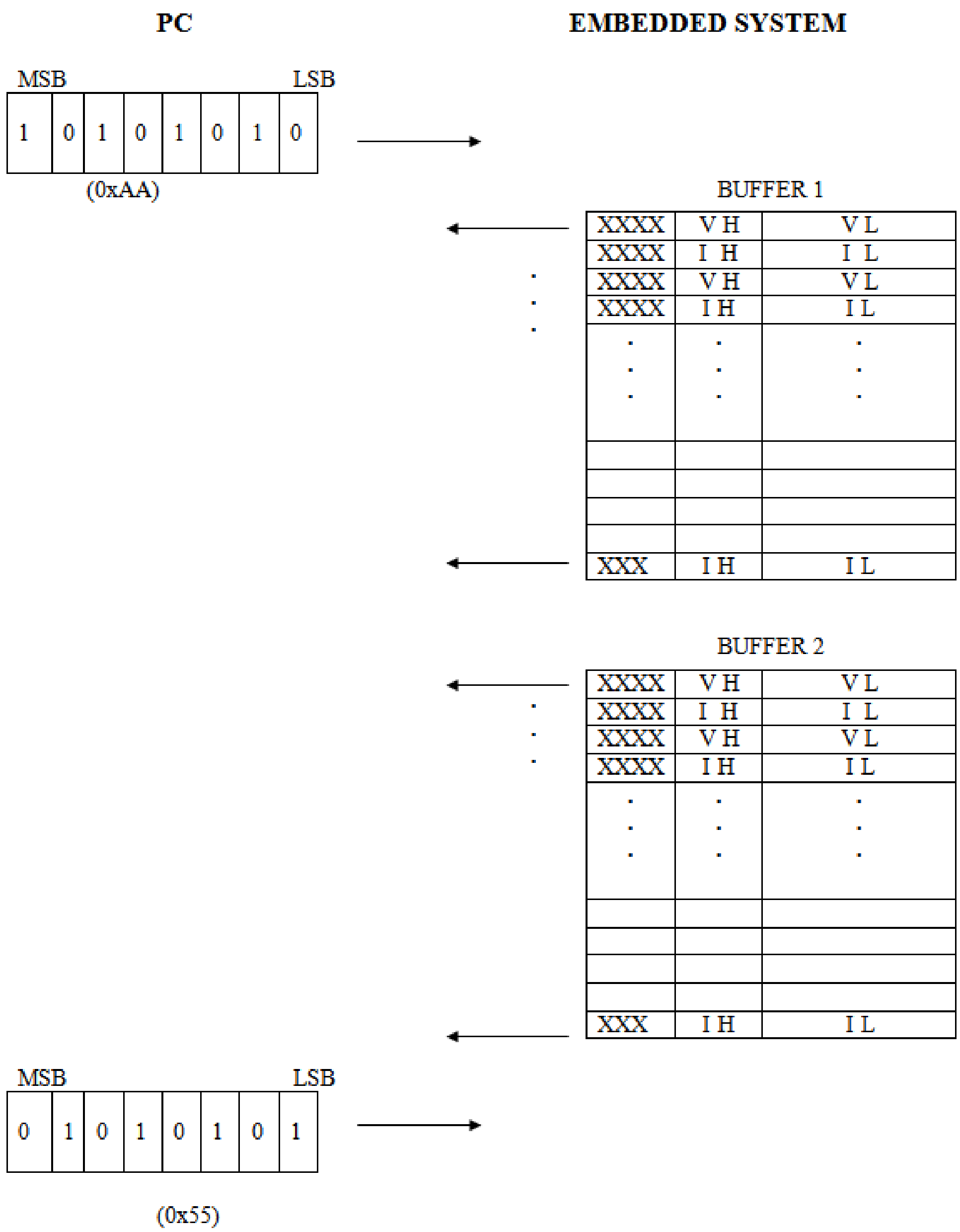


Figure 2.7 Structure of communication for the hardware processes.

Until the user clicks on the “stop” button, the system continues sending data to the PC and acquiring voltage and current data in the buffer structure. If the user pushes “stop” a new control word (0x55) is transmitted from the PC to the embedded system, and all processes (track, transmission and acquisition of A/D channels) are suspended.

Table 2.3 Transmission time.

Rate transmission	Byte transmission time	Buffer transmission time 2*32	Perturbation and storage time	Sampling time A/D
115200 bps	69.44 us	4.44 ms	160ms	1.017us

2.2.4 Software Design

Computer software written using MATLAB GUIDE was designed to provide a graphical user interface to control the maximum power point tracking. Figure 2.8 shows the graphical user interface, which consists of four areas. Area 1 shows the energy counter. The energy is measured in watts per hour (Wh). Area 2 is the power vs. time curve, calculated with voltage and current data from the PV module. Area 3 is the output voltage vs. time curve. Area 4 is the output current vs. time curve. Finally, the GUI includes the control buttons that allows the user to easily start and stop the system and reset the plots.

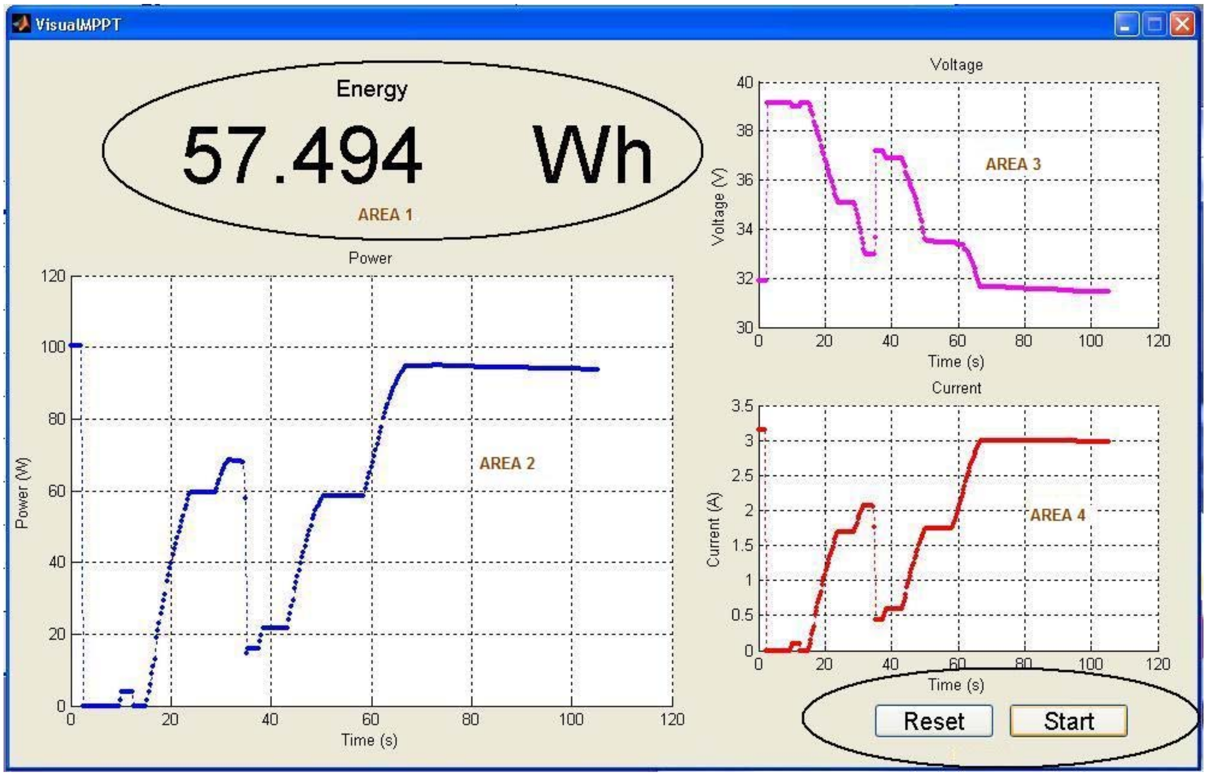


Figure 2.8 Graphic user interface.

Figure 2.9 shows the whole experimental maximum power point tracking system in operation. The system is connected to several measurement instruments. This setup was used to measure and characterize the performance of the maximum power point tracking at different temperature and solar radiation conditions.

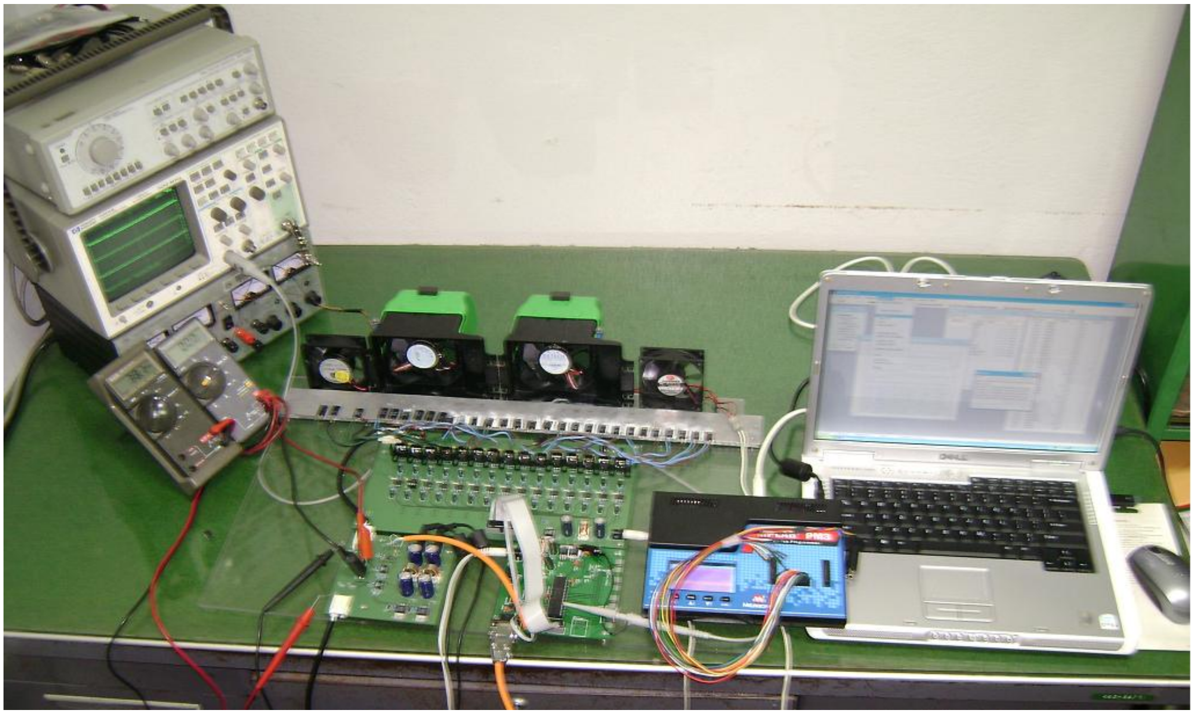


Figure 2.9 Maximum power point tracking in operation.

2.3 MPPT Methods Implemented for Testing

2.3.1 Improved Perturbation and Observation Method (MPPT IP&O)

The performance structure of the IP&O method is based on successive increases of load that allow the photovoltaic array to deliver the power required for the load. Figure 2.10 shows the dynamics of the IP&O tracking algorithm. Initially the panel is loaded with a high resistance and the power is measured. This is the comparative reference. Then, the photovoltaic array is loaded again, decreasing the resistance and measuring the power delivered. If this power is greater than the initial power, then the panel will be loaded with less resistance than the previous resistance. The process is repeated until the maximum power point is found, at which point the hysteresis system begins to operate. Section 2.3.3 explains the hysteresis system in greater detail.

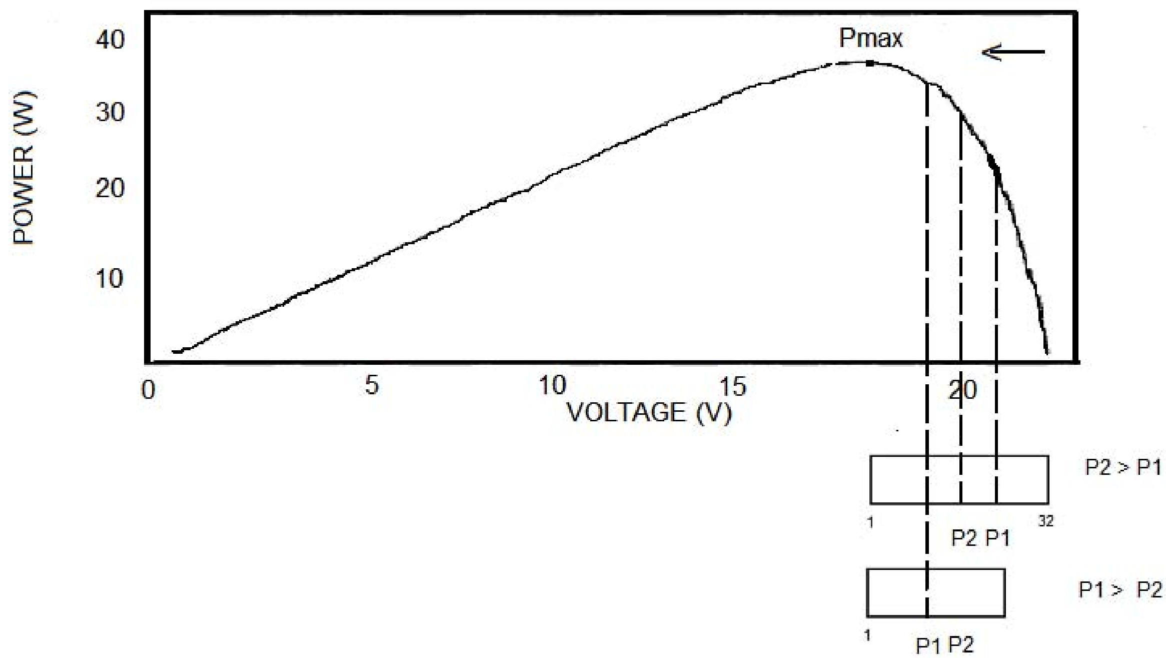


Figure 2.10 Improved Perturbation and Observation Method Structure.

Appendix A2 shows the Improved Perturbation and Observation Method (IP&O) flowchart. The firmware was written in Microchip Assembler Language based on this chart.

2.3.2 Selective Hopping Method (MPPT SH)

The dynamics of newly proposed method Selective Hopping are based on the power curve. The panel is loaded with the load P1 and is then loaded with the load P2 (see Figure 2.11). The powers are compared. If (P1 > P2) the extreme left of the curve is eliminated. On the other hand, if (P2 > P1) the extreme right of the curve is eliminated. When a part of the curve is eliminated the new test point is moved between the comparative point and its extreme. For example, in Figure 2.11 the (P2 > P1), therefore the left extreme of P1 is eliminated and the point P1 is moved between the point P2 and the right extreme. This process is repeated until the maximum power point is found. Figure 2.11 shows the overall dynamic of the tracking sequence.

Appendix A1 shows the Selective Hopping Method (SH) flowchart. The firmware was written in Microchip Assembler Language based on this flowchart.

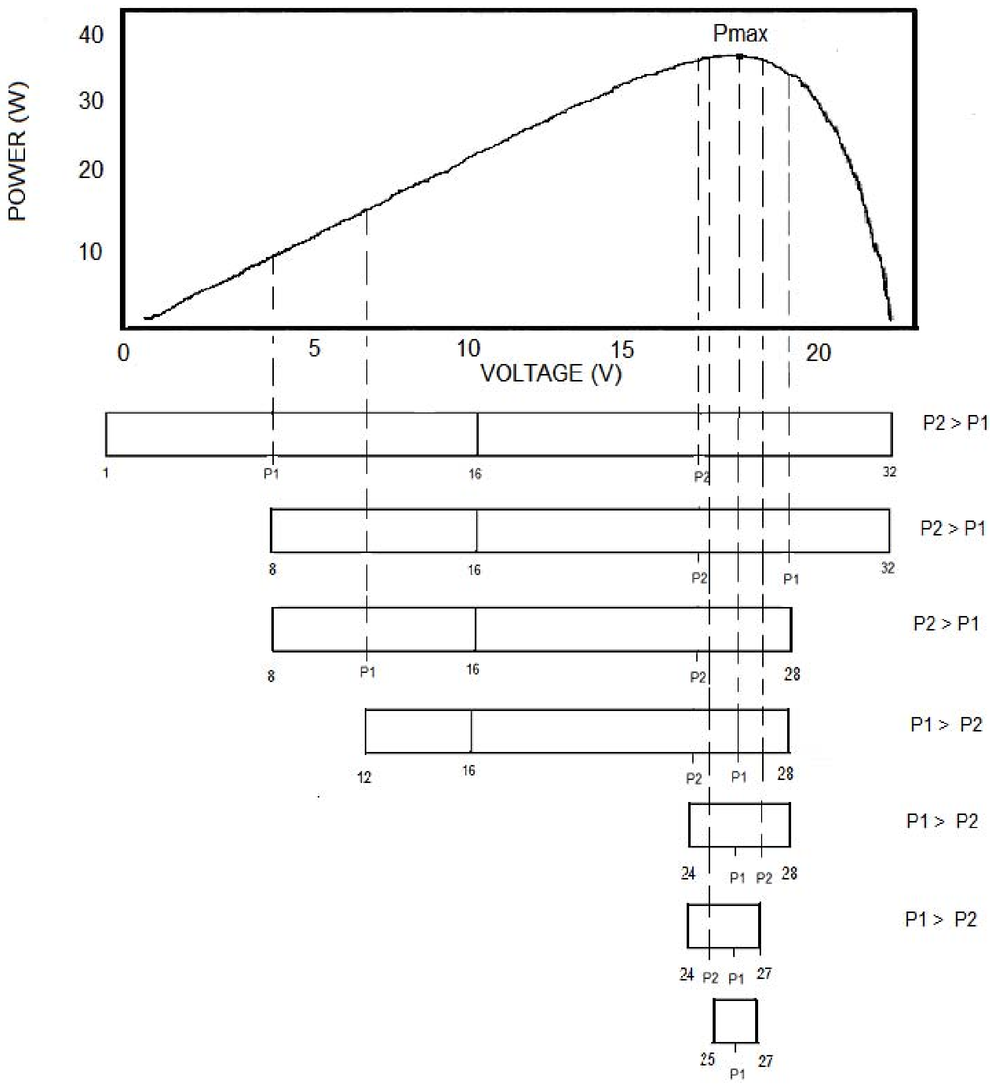


Figure 2.11 Selective Hopping Method Structure.

2.3.3 Hysteresis System

When the methods (SH or IP&O) find the maximum power point, a hysteresis system begins to work. This system loads the panel with less resistance and senses the amount of power being delivered to the load. If the power delivered is less than 3% of the nominal power of the photovoltaic array, the methods remain in the same point. If the power difference is between 3% and 5% a small area is tracked and a new point is found. If the difference exceeds 5% the systems reset and new overall tracking begins. The photovoltaic array (BP 3160) has a nominal power of 160W, where 3% and 5% are 5W and 8W, respectively. When the actual power sensed is greater than the previously sensed power, the system loads the panel with less resistance. This pattern is repeated until the maximum power point is found.

Both tracking methods have the same hysteresis system; the only difference between them is the tracking mechanism. Figure 2.12 shows the idea of a hysteresis system where Zone 1 does not track- it is the sensing zone. Zone 2 is a small tracking zone and Zone 3 is the reset zone.

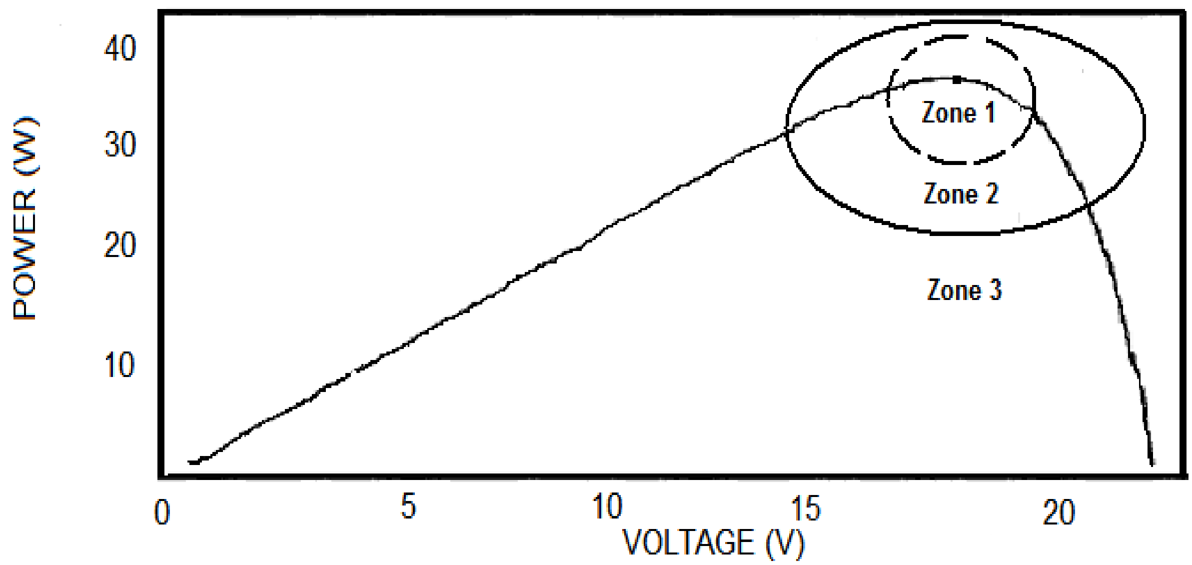


Figure 2.12 Structure of hysteresis system.

Appendix A2 shows the flowchart for the hysteresis system of the Improved Perturbation and Observation method and Appendix A1 shows the flowchart of the hysteresis system for the Selective Hopping Method.

3 RESULTS

The appropriate way to evaluate the performance of the tracking systems is using a controlled environment where temperature and radiation are controlled to assure similar conditions for the two tracking methods and allow for the behavior of the systems to be observed. Therefore a quasi-controlled environment was created with air conditioning (temperature control) and artificial illumination (radiation control) that allowed all characteristics of the systems to be evaluated. Figure 3.1 shows the quasi-controlled environment.

The tests did not allow relevant data to be obtained because the lamps used had lower power than needed and limited components (wavelengths) of solar radiation. Figures 3.2 and 3.3 show the results of the Improved Perturbation and Observation and Selective Hopping methods, respectively. Both measurements of power delivered for the photovoltaic array are negligible (<10mW) compared with the nominal power and represent approximately 0.05%, so these measurements were not taken in account for the analysis.

Because controlled environments are very expensive, the behavior of each tracking method was assessed in the natural environment. A test was designed to maintain similar conditions with respect to temperature and solar radiation. The test consists of observing the performance of one method and quickly changing methods to measure the performance of the next before testing conditions change. This test allowed for a comparative study between the two tracking methods.

Figure 3.1 Quasi-controlled environment for testing both tracking methods.

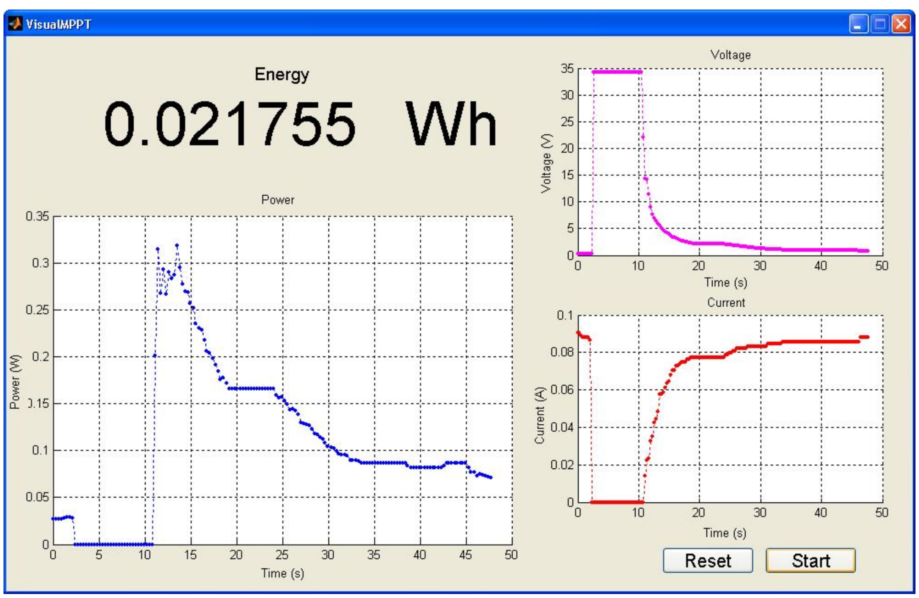


Figure 3.2 Test of IP&O method in the quasi-controlled environment.

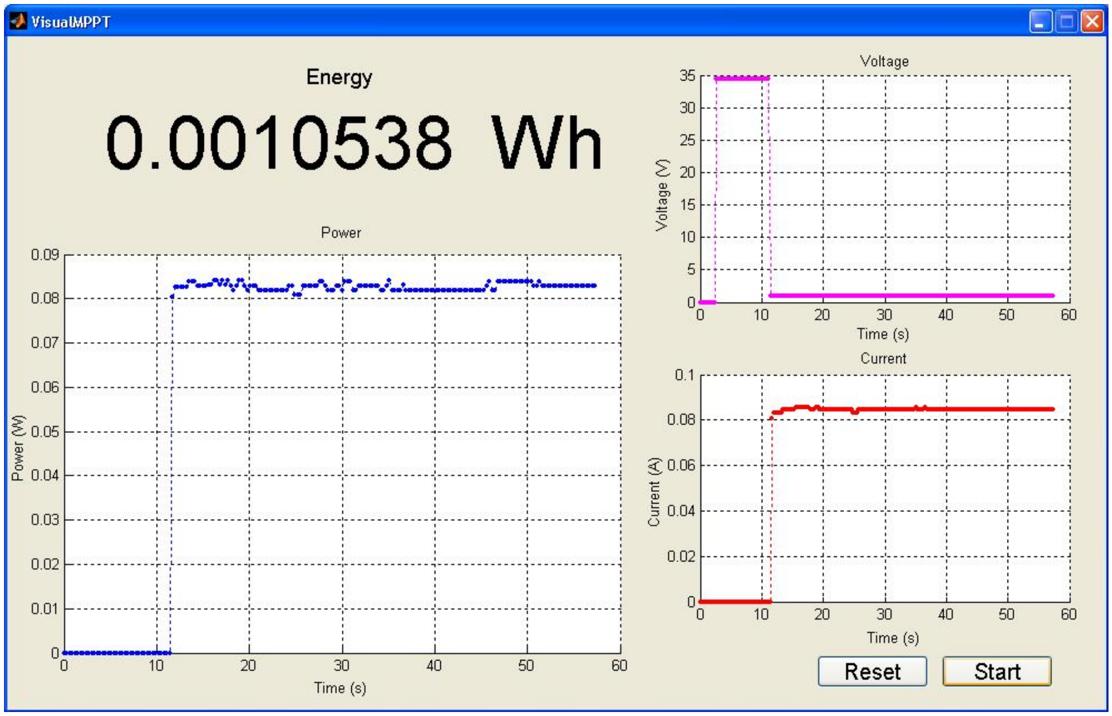


Figure 3.3 Test of SH method in the quasi-controlled environment.

3.1 Performance characteristics of the IP&O Method

Figure 3.4 shows a simple tracking using the IP&O method. The power vs. time chart shows a curve that rises continuously with respect to time until the maximum power point is found at approximately 115W. The variations of current are also shown, from the open-circuit current to the maximum current delivered by the photovoltaic array (3.4A). The voltage vs. time curve shows the variations from open-circuit voltage to maximum power point voltage (32V), where the PV can deliver the maximum current.

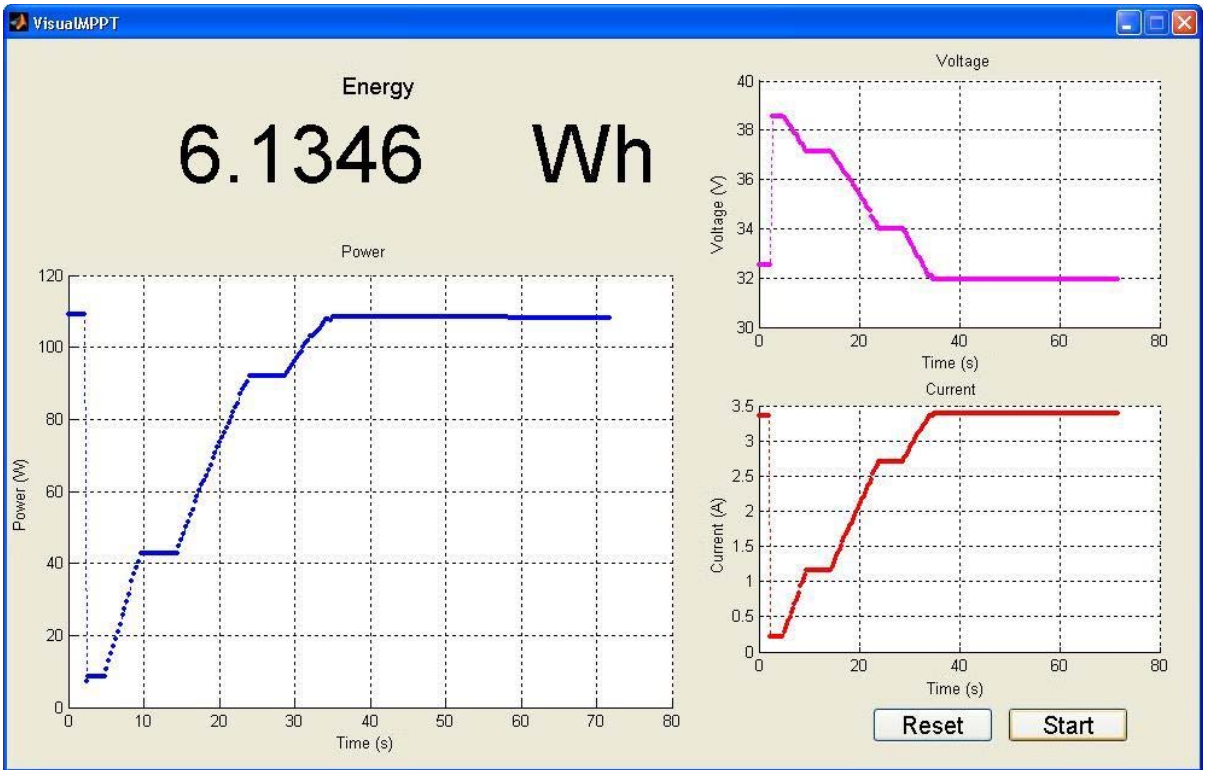


Figure 3.4 Simple tracking of maximum power point with IP&O.

Figure 3.5 shows the dynamic of the power, current, and voltage when sudden variations in solar radiation happen. The tracking algorithm always finds the maximum power point. This is observed on the power vs. time plot around 50s, 85s and 110s. Figure 3.6 shows three successive tracking tests carried out in a short period of time using the control buttons. It shows that the convergence point is equal, approximately 115W for each, and that the algorithm does not find a local maximum power.

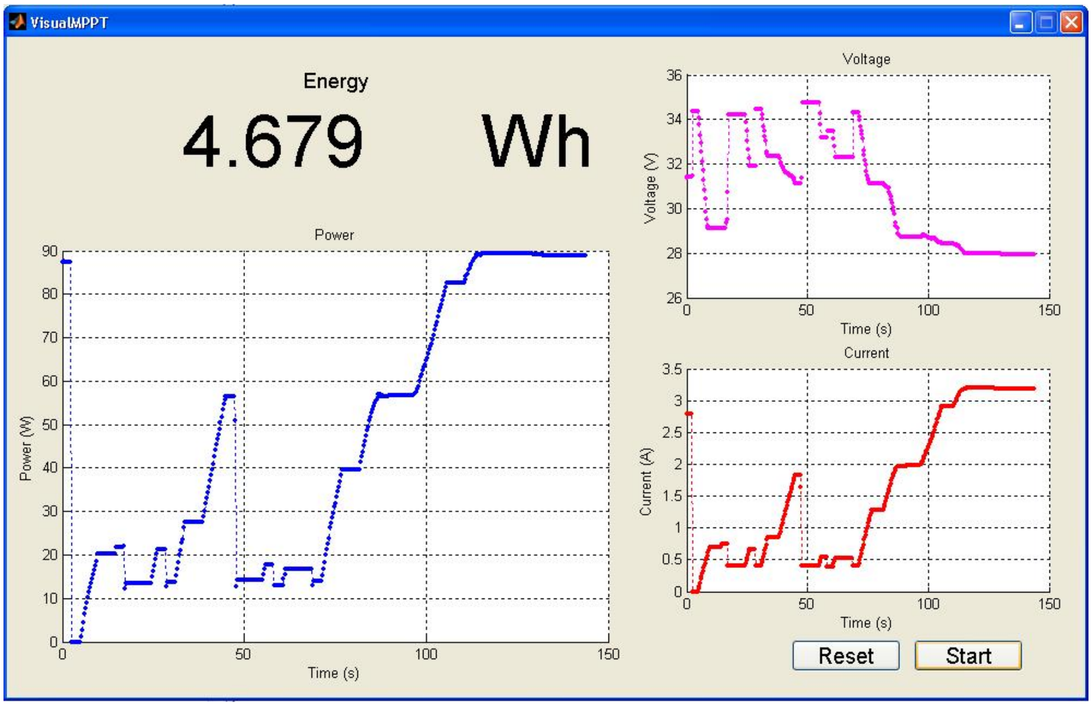


Figure 3.5 IP&O System Dynamics.

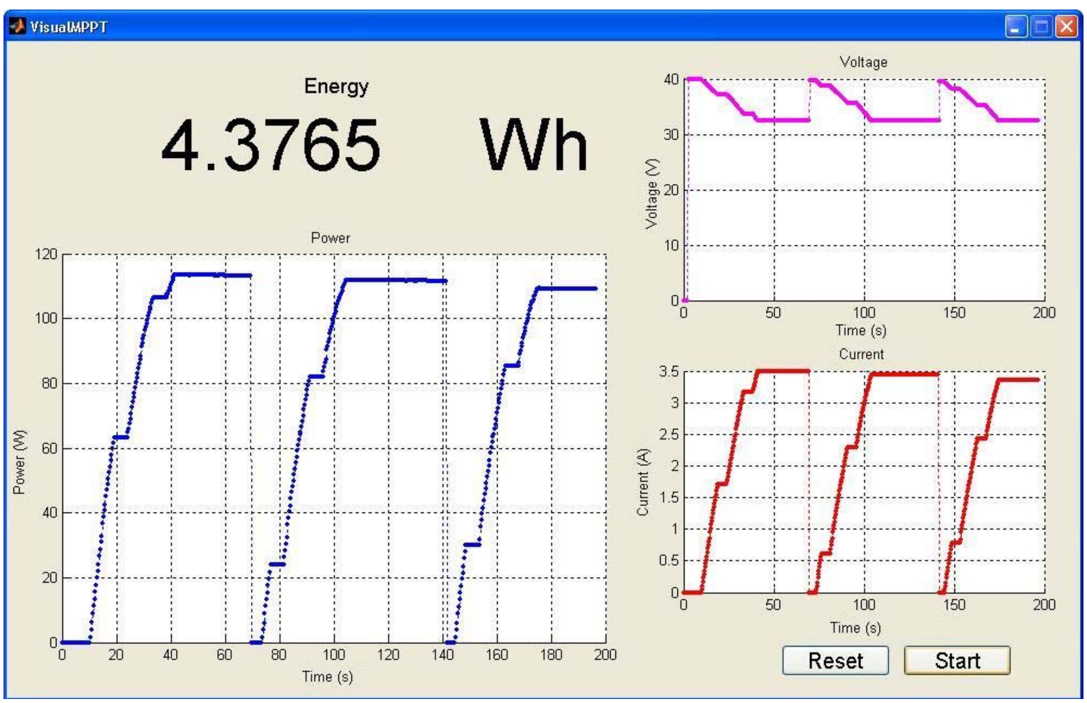


Figure 3.6 Successive tracking tests of IP&O method.

Figure 3.7 shows the performance of the IP&O method in the presence of intense variations of solar radiation. For this test, the photovoltaic array is manually covered and then uncovered, allowing a change from zero solar radiation to the maximum available radiation to be observed. The graph shows that the IP&O method returned to the maximum power point after adjusting to these radical variations at approximately 90W. The method did not find a local maximum power. It is important to note that these conditions are not possible in practice and the test was performed to observe the reaction of the methods to extreme changes in radiation. This note applies to tests performed on the SH method as well.

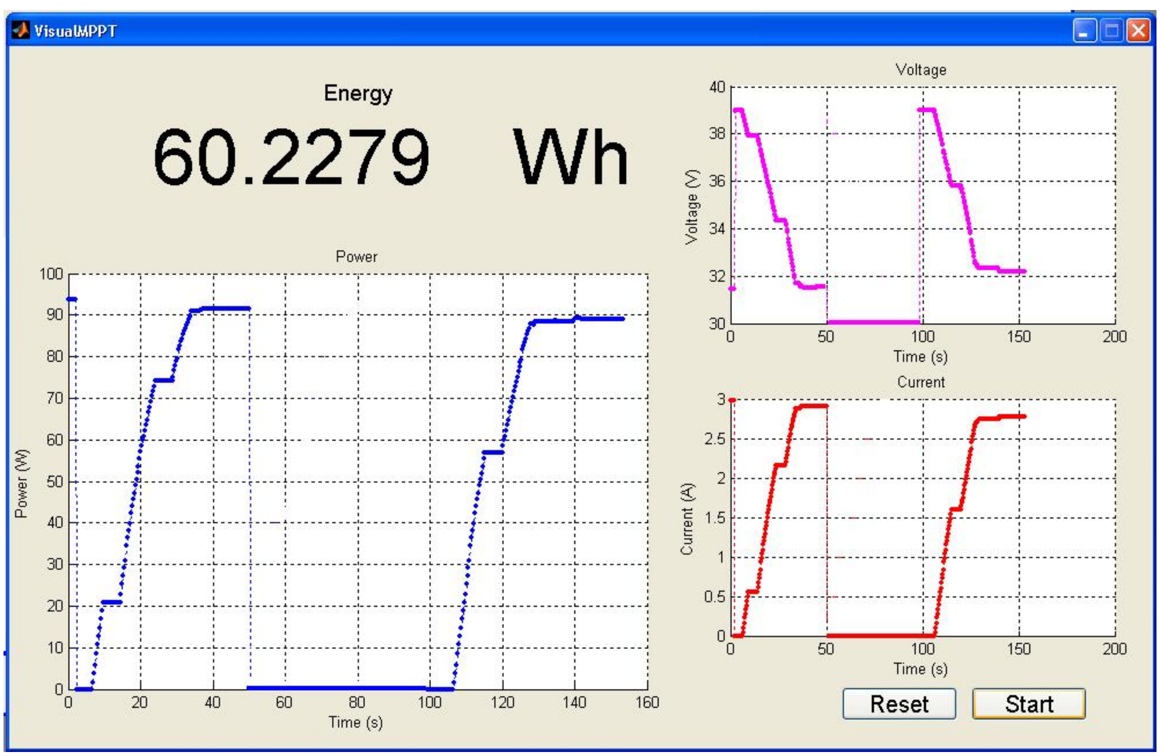


Figure 3.7 Recovery of maximum power point after drastic changes- SH.

3.2 Performance characteristics of the SH Method

Figure 3.8 shows a simple tracking test using the SH method. The power curve shows a rapid change with respect to time from zero power to maximum power, in contrast with the continuously rising curve observed with the IP&O method. It is also observed that the SH method had fewer iterations than the IP&O method (shown in Figure 3.4), making SH more efficient computationally than IP&O. Rapid variation of the current is also seen from open-circuit current to maximum current delivered. The voltage vs. time curve shows the variations of the tracking dynamic (open circuit voltage – maximum power voltage).

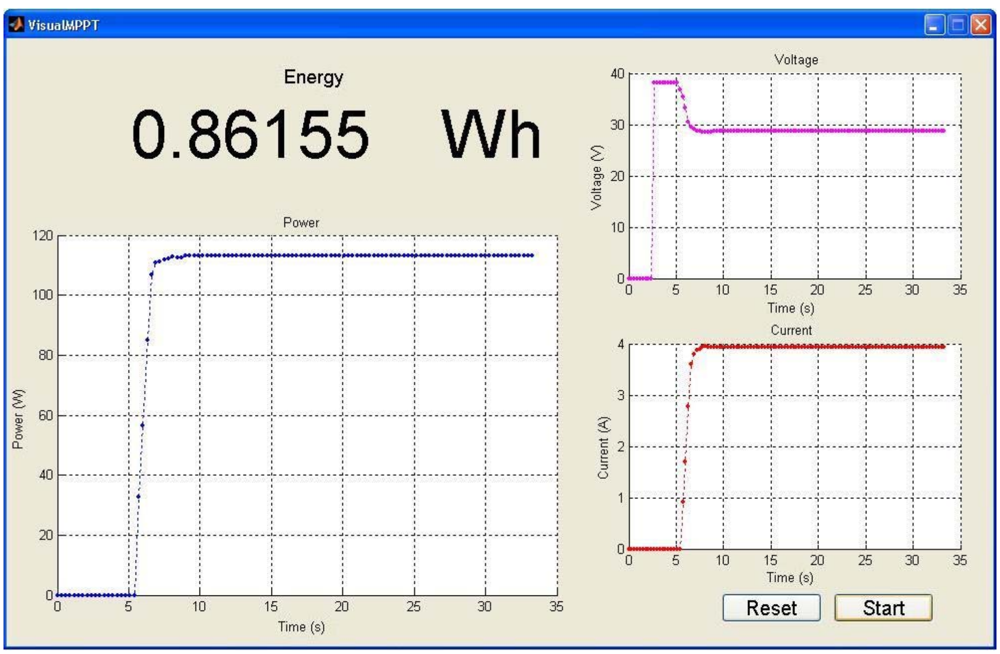


Figure 3.8 Simple tracking of maximum power point - SH.

Figure 3.8 shows that the SH method has a shorter convergence time than seen with the IP&O method in Figure 3.4 (2s SH vs. 30s IP&O). The power levels in the two figures are similar (117W SH vs. 115W IP&O).

Figure 3.9 shows the behavior of the system when there is an increase in solar radiation. This graph illustrates the tracking dynamic for finding the maximum power point.

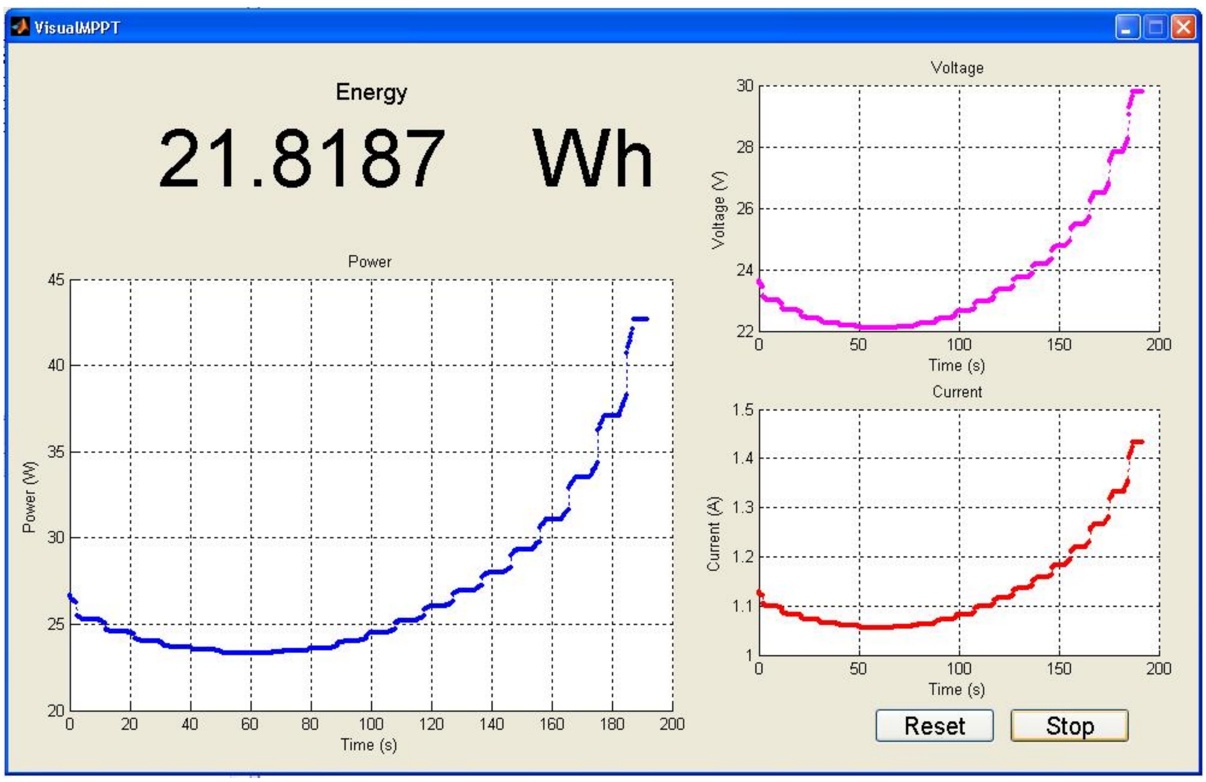


Figure 3.9 System dynamic - SH.

Figure 3.10 shows a sequence of successive tracking tests carried out in a short period of time. It shows that the SH method converges in the same point for each sequence. The performance of this method is similar to that of IP&O and avoids finding local maximum power. However, the SH always returns to the same maximum (see Figure 3.10), while there is a small decrease in the IP&O (as seen in Figure 3.6). These tests also show the convergence speed of the SH versus that of the IP&O.

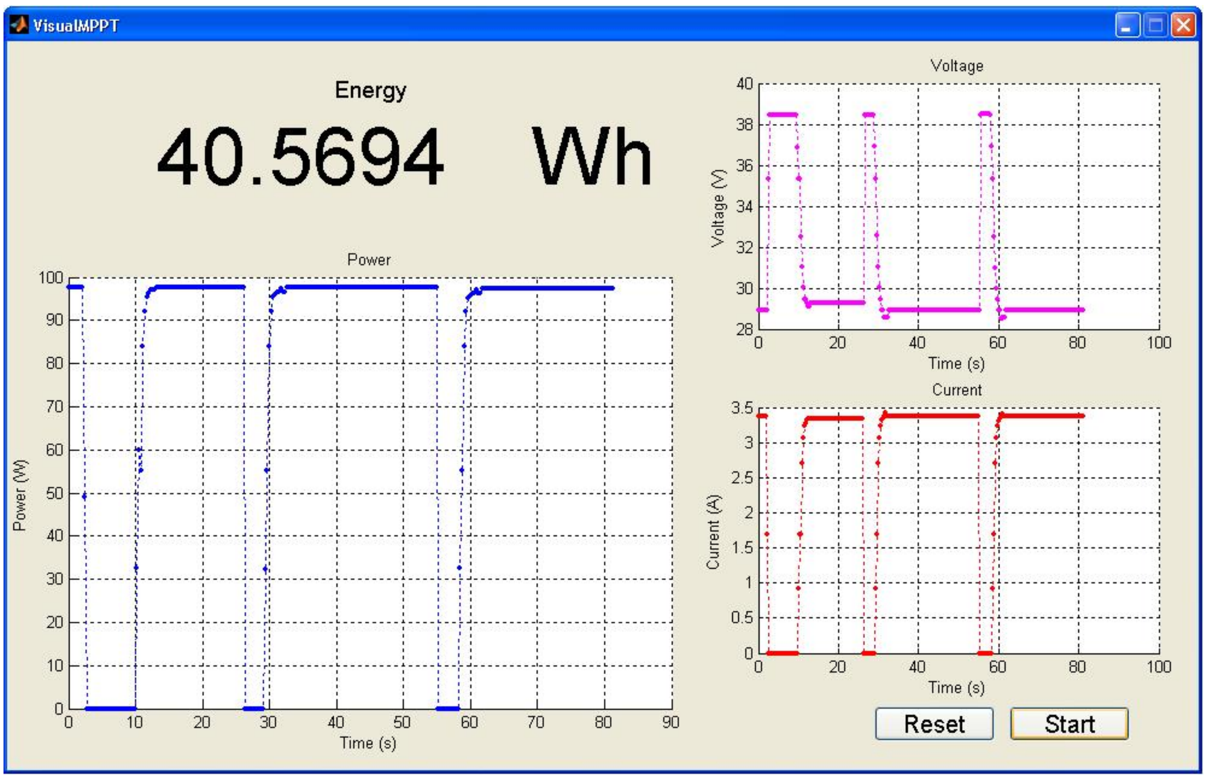


Figure 3.10 Successive tracking tests of SH method.

Figure 3.11 registers the behavior of the SH method in the presence of sudden variations in solar radiation. This test was carried out in the same manner as the tests for the IP&O method in Figure 3.7. The graph shows that the SH method did not return to the maximum power point after these variations in radiation. The SH method finds a local maximum power, demonstrating a problem with the method. However, as stated before, this problem has little relevance, since these drastic variations do not occur in practical applications.

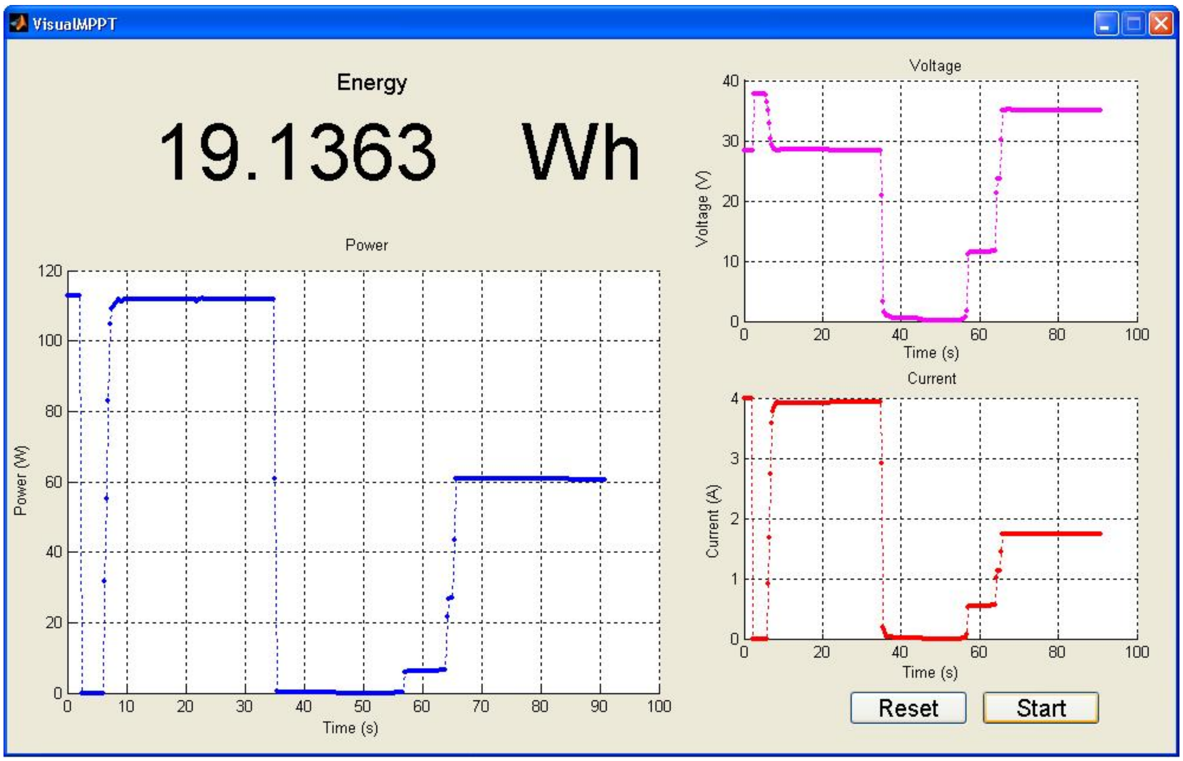


Figure 3.11 Successive tracking tests of the SH method.

3.3 Comparative Performance between IP&O and SH Methods

In the comparative analysis of the SH and IP&O methods multiple tests were carried out to observe the performance of the algorithms in similar conditions and to obtain a base for a comparative study and complete characterization of the methods. The tests initially used one tracking method and a short time (approximately 20s). Later a new tracking method was programmed. The short interval between the tests assured that both were carried out under similar conditions with respect to temperature and solar radiation. Table 3.1 contains the most relevant data for convergence time and available power when the SH method was programmed first, followed by IP&O. Table 3.2 contains data derived by testing IP&O first, followed by the SH method.

Table 3.1 SH - IP&O Sequence.

Test Name	Convergence Time (s)		Available Power (W)	
	SH	***IP&0***	***SH***	***IP&0***
TS4	2	26	88	86
TS5	1	20	91	86
TS10	2	31	119	112
TS11	2	32	117	107
TS13	3	33	106	100
TS14	3	29	105	95
TS15	3	29	102	96
TS16	2	24	100	95
TS18	2	20	96	92
TS20	2	23	85	82

For example, Test TS5 of Table 3.1 is shown in Figure 3.12 (corresponding sequence SH – IP&O) where the minimum convergence time of the IP&O method was registered. The results are a convergence time of 1s for the SH method and 20s for the IP&O method. In terms of power output the SH provided 91W while the IP&O provided 86W. Test TS13 of Table 3.1 is shown in Figure 3.13; maximum convergence time for the IP&O method was 33s and 3s for the SH method. It is shown that power output was 106W and 100W for the SH and for the IP&O, respectively. Figures 3.12 and 3.13 show that when the photovoltaic array delivers more power, the algorithm of the IP&O method needs more time to converge while the algorithm of SH method needs the same convergence time independent of the power delivered for the PV.

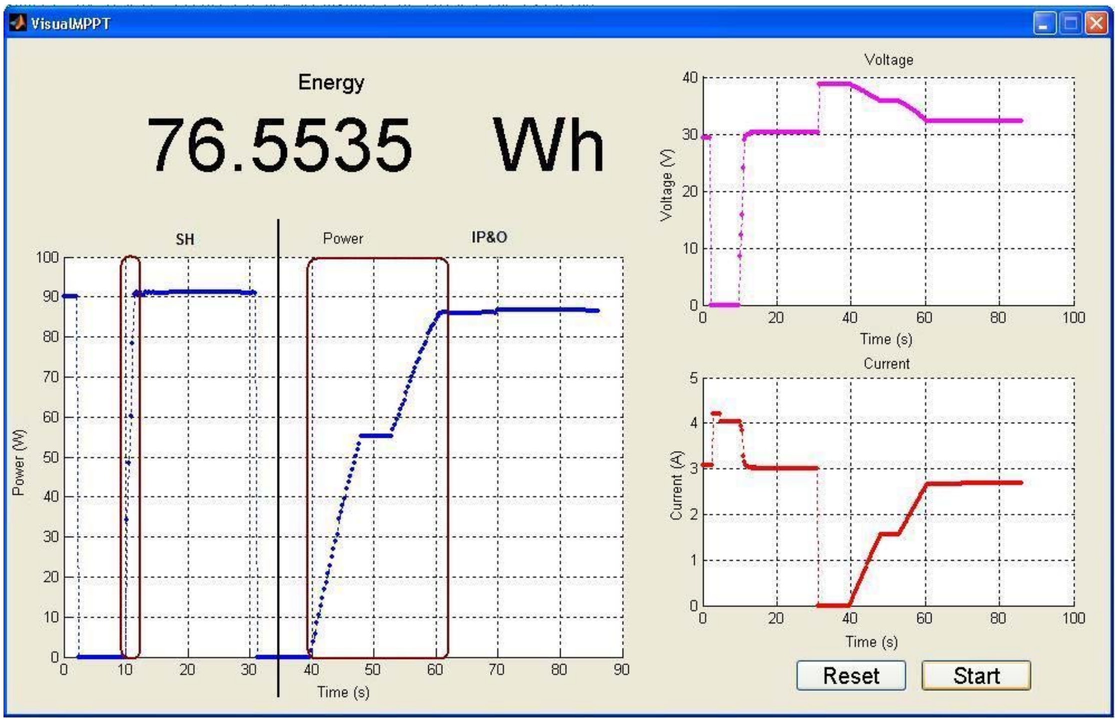


Figure 3.12 Test TS5 SH – IP&O Sequence.

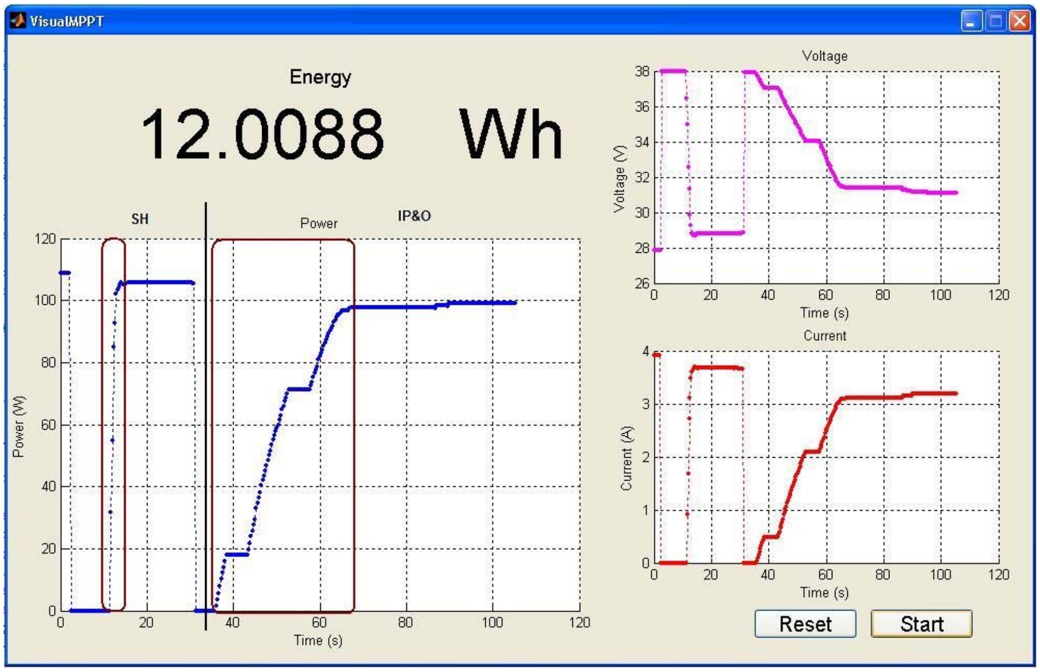


Figure 3.13 Test TS13 SH – IP&O Sequence.

Table 3.2 IP&O – SH Sequence.

Test Name	**Convergence Time**		**Available Power (W)**	
	IP&0	***SH***	***IP&0***	***SH***
TS	31	2	108	117
TS7	30	2	110	114
TS9	30	2	101	109
TS11	32	2	104	111
TS12	32	2	104	110
TS13	31	2	98	104
TS14	28	2	96	100
TS16	22	2	85	88
TS18	21	2	82	84
TS20	21	3	78	78

Figure 3.14 shows the results of Test TS11 from Table 3.2 (corresponding sequence IP&O - SH) where the maximum convergence time of the IP&O method was registered. In the IP&O method a convergence time of 32s was registered, and the convergence time for the

SH method was 2s. Figure 3.15 shows the results of Test TS18 of the same sequence. In this case the minimum convergence time registered for the IP&O method was 21s and 2s for the SH method, and the power output was 82W and 84W for the IP&O and SH methods, respectively. In the IP&O method, when the PV delivers more power the convergence time is greater, while for the SH method a constant convergence time is observed independent of the power delivered.

In most tests performed the SH method is at least 10 times faster than IP&O. The SH method had almost the same convergence time independent of power available from the PV, while the convergence time of IP&O had a behavior proportional to the available power for the PV.

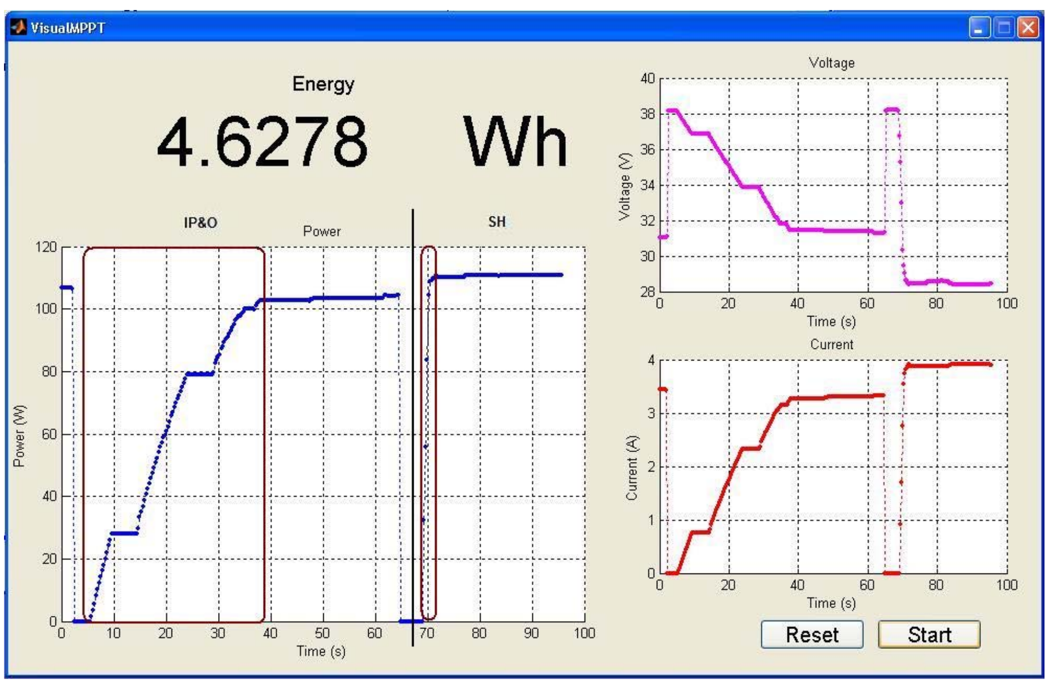


Figure 3.14 Test TS11 IP&O-SH Sequence.

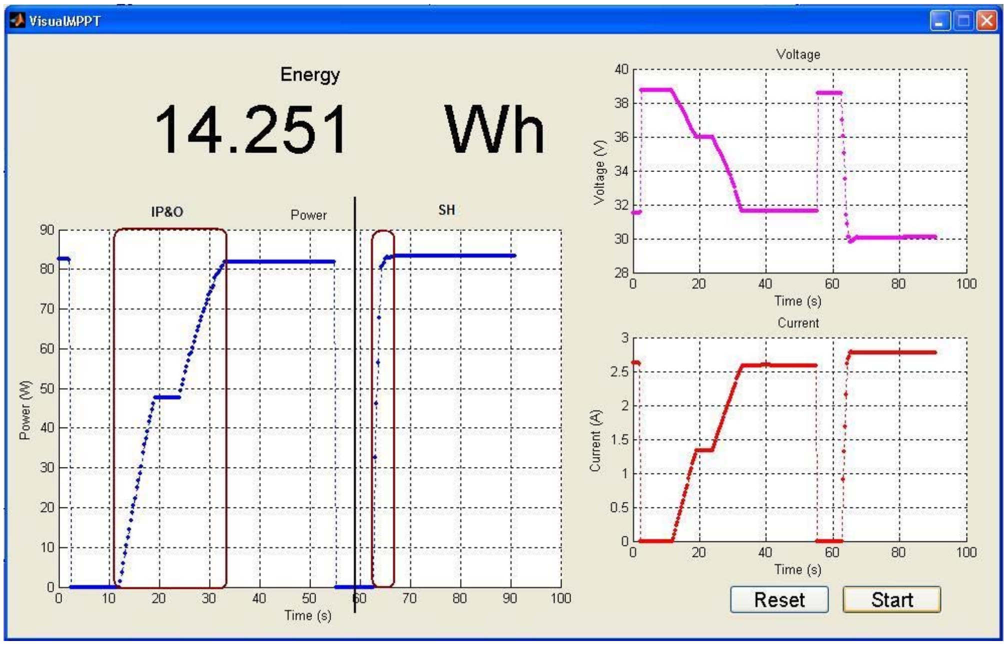


Figure 3.15 Test TS18 IP&O-SH Sequence.

Figures 3.16 and 3.17 show tests TS10 and TS20 from Table 3.1 (corresponding to the sequence SH – IP&O). Figure 3.16 shows the maximum power available with the SH method and the difference of power between the SH – IP&O methods. This graph shows that the SH method can find a higher maximum power point that allows more power to be delivered from the PV than with the IP&O method. This is important, since more power in time means more energy available. Figure 3.17 shows the minimum power available in the tests with the SH method. The SH method consistently extracts more power than the IP&O method, although the difference between the methods decreases when there is less available power. The power difference range registered in the tests was from 3- 10W in the SH - IP&O sequence. Taking all tests into consideration, the SH delivered on average 5% more power than the IP&O. Figure 3.18 shows the maximum power points in both methods from table 3.1 in the SH-IP&O sequence. The figure clearly shows that SH extract more power from the PV panel during such testing sequence.

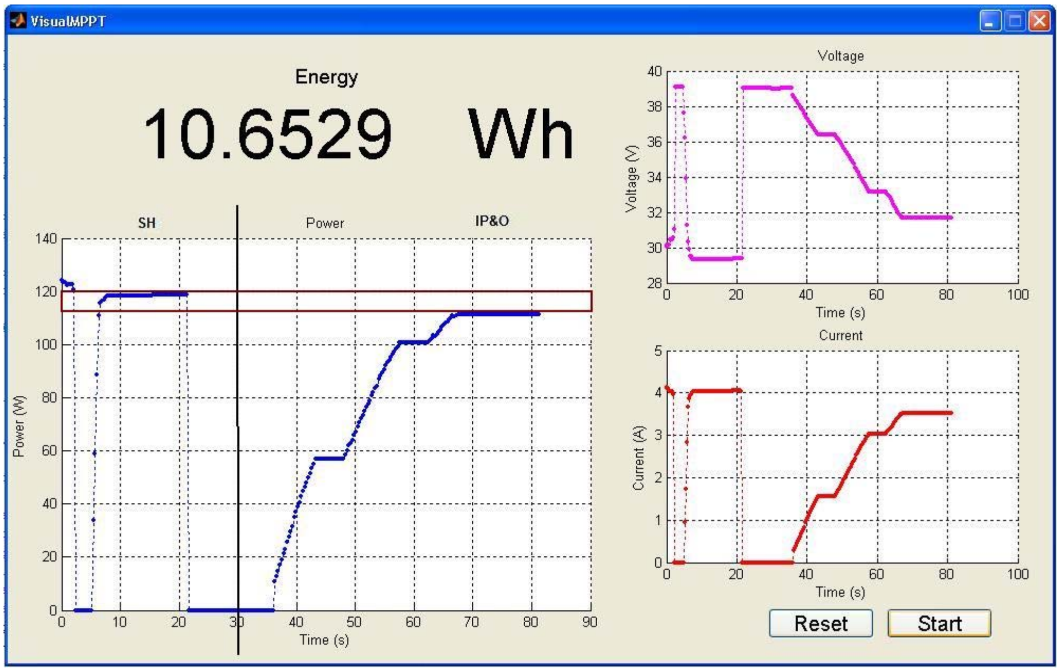


Figure 3.16 Test TS10 SH – IP&O Sequence.

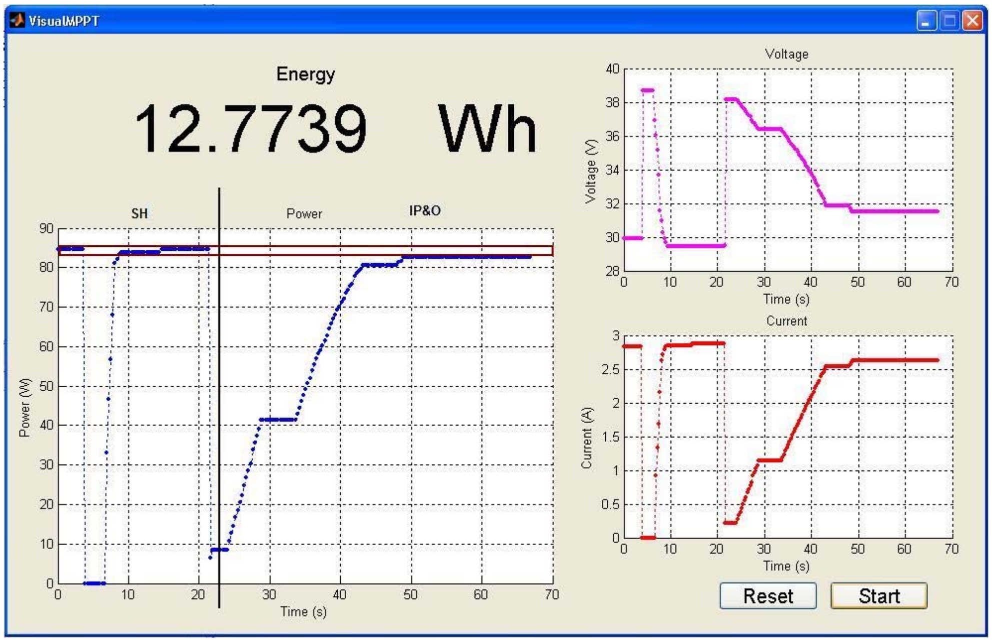


Figure 3.17 Test TS20 SH – IP&O Sequence.

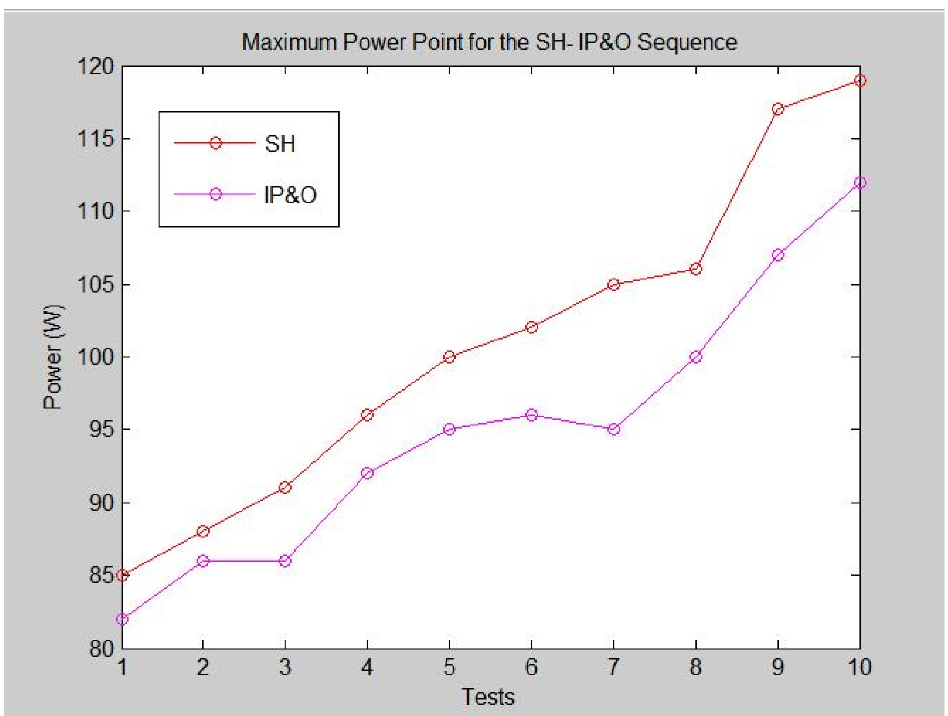


Figure 3.18 Maximum power point for the SH - IP&O Sequence.

Figures 3.19 and 3.20 show tests TS7 and TS20 of Table 3.2 (corresponding to the sequence IP&O - SH). Figure 3.19 shows the maximum power available with the SH method for this sequence and the power difference between the SH and IP&O methods. This sequence has similar results to those of the SH-IP&O sequence. The SH method continues to find a higher maximum power point than the IP&O method.

Figure 3.20 shows the minimum power available in the tests with the SH method. The difference in power between the methods is reduced. This condition is similar for both sequences and the power difference range between the IP&O and SH methods for all tests realized in this sequence is 0-8W. Figure 3.21 shows the maximum power points in both methods from table 3.2, regarding the IP&O SH sequence. The same response as in figure 3.18 is shown in this figure, the SH extract more power from the PV panel.

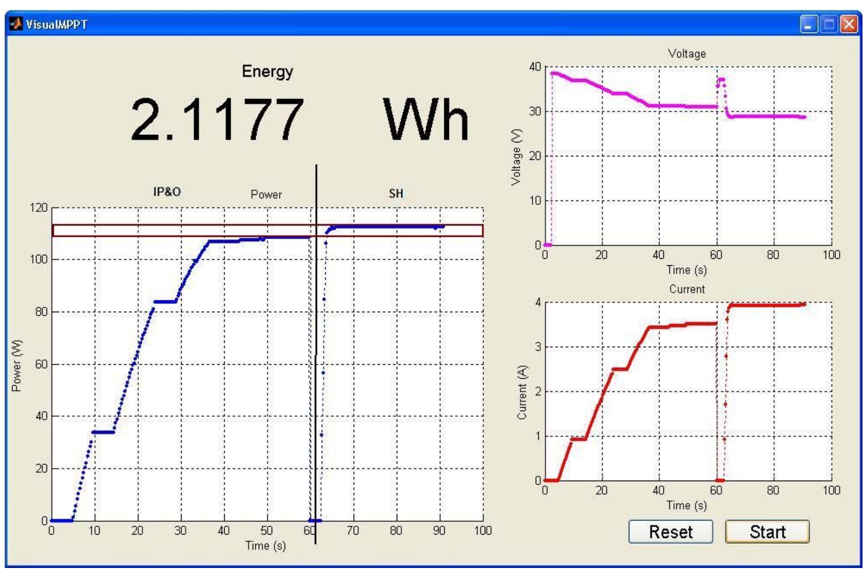


Figure 3.19 Test TS7 IP&O – SH Sequence.

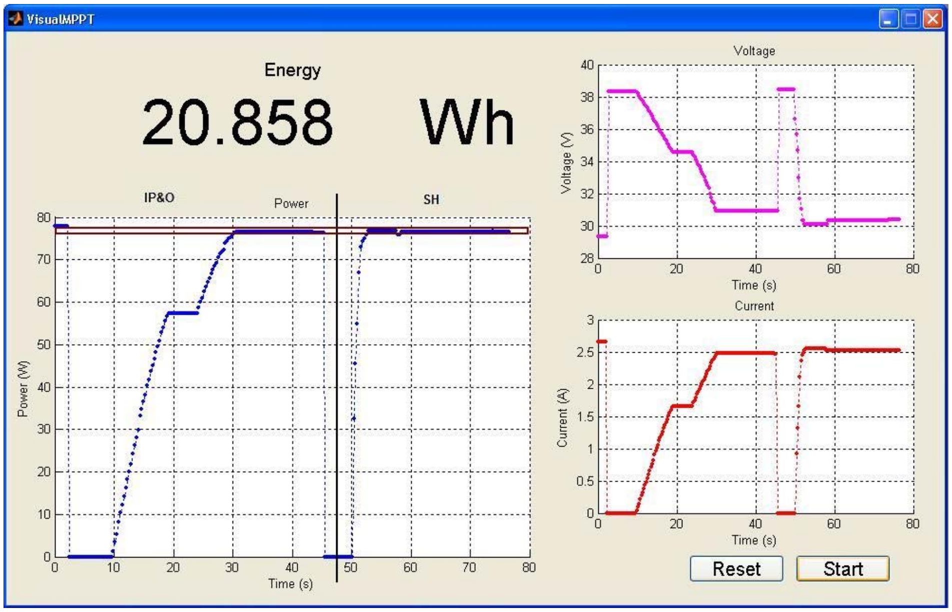


Figure 3.20 Test TS20 IP&O – SH Sequence.

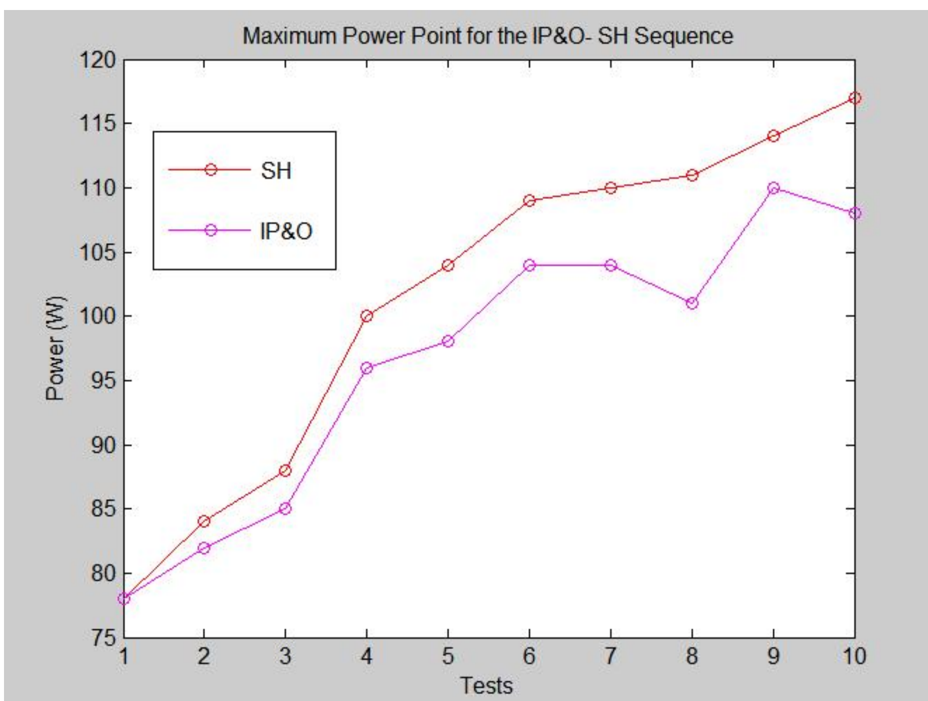


Figure 3.21 Maximum power point for the IP&O-SH Sequence.

In all tests realized in both sequences, the SH method found a greater maximum power point than the IP&O method. The power difference between the methods is smaller when the available power for the photovoltaic array decreases. The SH method also finds the maximum point of operation in a shorter amount of time.

The problem with the SH method is that local maximum is found when the radiation changes suddenly (maximum-zero-maximum). This problem does not represent a threat to the practical application of the method because these extreme conditions are not present in nature.

The results presented in this chapter demonstrate that the SH method is much faster than the IP&O method. Therefore, the ability to recover the maximum power point in the presence of minor variations in radiation must also be faster. However, the differences between the methods cannot be quantified using the results of the tests that have been carried out so far because a controlled environment is the only way to guarantee that the variations in radiation are the same when testing the two systems.

4 CONCLUSIONS AND FUTURE WORK

A maximum power point tracking system using a modified version of the P&O method was developed to evaluate power point tracking methods. Two tracking algorithms were designed based on the Perturbation and Observation method. The Improved Perturbation and Observation and the Selective Hopping methods were fully characterized with a series of tests. The tests allowed the behavior of the methods to be observed with respect to the convergence, dynamic performance and variations in solar radiation. The results showed that the tracking methods improve upon some of the constraints reported with previous methods. A low computational cost is conserved and the hysteresis system allows the power deflection to be decreased for both tracking methods. The tests also showed that with the IP&O method there is no problem with convergence, while the SH method found maximum local power when the photovoltaic array was subjected to drastic changes in solar radiation levels; however, this condition is not likely to occur in nature.

The tests for the two methods show that the SH method has a shorter convergence time, in one order of magnitude, than that of the IP&O method. This result is independent of the power level available to the photovoltaic array. On the other hand, the IP&O method has a greater convergence time when there is more power available. On the tests performed it was observed that the SH method allows the PV module to deliver more power to the load than the IP&O method.

The tracking system outlined here optimizes the efficiency of solar panels, decreasing the number of panels necessary to provide the energy needed and thereby decreasing the overall cost of a PV power system. An important potential application of the method is the creation of a renewable energy system that can be integrated with Off-the-Grid radars.

In order to accomplish this, a DC-DC converter that uses the Selective Hopping tracking system must be designed. This method can also be used to study the amount of energy available in a given area in order to validate previously collected solar radiation data. These studies will provide more accurate information, which will assist in improving the designs of renewable energy systems. The system can be used as a tool to evaluate the performance of photovoltaic arrays other than the one used here. It can also be implemented with DC/AC inverters for applications in residential, commercial and industrial PV systems. Finally, the system can be used as an evaluation and characterization mechanism for other tracking systems based on the electrical characteristics of photovoltaic arrays.

REFERENCES

A. Al-Amoudi and L. Zhang, "Optimal control of a grid-connected PV system for maximum power point tracking and unity power factor," in *Proc. Seventh Int. Conf. Power Electron. Variable Speed Drives*, 1998, pp. 80–85.

Bernard Bekker, H.J.Beukes, "Finding an Optimal PV Panel Maximum Power Point Tracking Method", *AFRICON, 2004. 7th AFRICON Conference in Africa Volume 2, 2004 Page(s):1125 - 1129 Vol.2.*

Chee Wei Tan; Green, T.C.; Hernandez-Aramburo, C.A.; "An Improved Maximum Power Point Tracking Algorithm with Current-Mode Controlfor Photovoltaic Applications" *Power Electronics and Drives Systems, 2005. PEDS 2005. International Conference on Volume 1, 16-18 Jan. 2006 Page(s):489 – 494.*

D. J. Patterson, "Electrical system design for a solar powered vehicle," in *Proc. 21st Annu. IEEE Power Electron. Spec. Conf.*, 1990, pp. 618–622.

Dong-Yun Lee; Hyeong-Ju Noh; Dong-Seok Hyun; Ick Choy; "An improved MPPT converter using current compensation method for small scaled PV-applications", *Applied Power Electronics Conference and Exposition, 2003. APEC '03. Eighteenth Annual IEEE Volume 1, 9-13 Feb. 2003 Page(s):540 - 545 vol.1.*

Femia, N.; Petrone, G.; Spagnuolo, G.; Vitelli, M.; Power Electronics, "Optimization of perturb and observe maximum power point tracking method ", *IEEE Transactions on Volume 20, Issue 4, July 2005 Page(s):963 - 973 .*

Femia,N.; Petrone, G.; Spagnuolo, G.; Vitelli, M.; "Increasing the efficiency of P&O MPPT by converter dynamic matching", *Industrial Electronics, 2004 IEEE International Symposiumon Volume 2, 4-7 May 2004 Page(s):1017 - 1021 vol. 2.*

G. W. Hart, H. M. Branz, and C. H. Cox, "Experimental tests of open loop maximum-power-point tracking techniques," *Solar Cells*, vol. 13, pp. 185–195, 1984.

H.-J. Noh, D.-Y. Lee, and D.-S. Hyun, "An improved MPPT converter with current compensation method for small scaled PV-applications," in *Proc. 28th Annu. Conf. Ind. Electron. Soc.*, 2002, pp. 1113–1118.

Joe-Air Jang,Tson-Liang Huang, Ying-Tung Hsiao, Chia-Hong Chen, ", Maximum Power Tracking for Photovoltaic Power Systems", *Tamkang Journal of Science and Engineering, Vol 8, N 2, pp147-153 (2005).*

J. J. Schoeman and J. D. vanWyk, "A simplified maximal power controller for terrestrial photovoltaic panel arrays," in *Proc. 13th Annu. IEEE Power Electron. Spec. Conf.*, 1982, pp. 361–367.

Jorge M. Trabal, Brian C. Donovan, Manuel Vega ,Victor Marrero, David J. McLaughlin, and José G. Colom, "Puerto Rico Student Test Bed Applications and System Requirements Document Development", *9th International Conference on Engineering Education July 23 – 28, 2006.*

K. Kobayashi, H. Matsuo, and Y. Sekine, "A novel optimum operating point tracker of the solar cell power supply system," in *Proc. 35th Annu. IEEE Power Electron. Spec. Conf.*, 2004, pp. 2147–2151.

K. Samangkool, S. Premrudeepreechacharn, "Maximum Power Point Tracking Using Neural Networks for Grid-Connected Photovoltaic System". *Future Power Systems, 2005 International Conference on 16-18 Nov. 2005 Page(s):1 – 4.*

L. Zhang, Y. Bai, and A. Al-Amoudi, "GA-RBF neural network based maximum power point tracking for grid-connected photovoltaic systems," in *Proc. Int.Conf. Power Electron.,Machines and Drives*, 2002, pp. 18–23.

M. A. S. Masoum, H. Dehbonei, and E. F. Fuchs, "Theoretical and experimental analyses of photovoltaic systems with voltage and current-based maximum power-point tracking," *IEEE Trans. Energy Convers.*, vol. 17, no. 4, pp. 514–522, Dec. 2002.

N. Mutoh, T. Matuo, K. Okada, and M. Sakai, "Prediction-data-based maximum-power-point-tracking method for photovoltaic power generation systems," in *Proc. 33rd Annu. IEEE Power Electron. Spec. Conf.*, 2002, pp. 1489–1494.

Manuel A. Vega, José G. Colom, Lionel Orama "Student Developed Meteorological Radar Network for Western Puerto Rico", *The 49th IEEE International Midwest Symposium on Circuits and Systems. San Juan, Puerto Rico, from August 6 to 9, 2006.*

M. Buresch, *Photovoltaic Energy Systems*. New York: McGraw Hill,1983

McLaughlin D., "CASA Strategic Plan", 2003.
Available from http://www.casa.umass.edu/.

M.-L. Chiang, C.-C. Hua, and J.-R. Lin, "Direct power control for distributed PV power system," in *Proc. Power Convers. Conf.*, 2002, pp. 311–315.

Minwon Park, IN-Keun Yu, "A Study on the Optimal Voltage for MPPT obtained by Suface Temperature of Solar Cell ", *The 30th Annual Conference of the IEEE Industrial Electronics Societ, November 2-6, 2004, Busan Korea.*

N. Kasa, T. Iida, and H. Iwamoto, "Maximum power point tracking with capacitor identifier for photovoltaic power system," in *Proc. Eighth Int.Conf. Power Electron. Variable Speed Drives*, 2000, pp. 130–135.

Nobuyoshi Mouth, Masahiro Ohno, Takayoshi Inoue, "A Method for MPPT Control While Searching for Parameters Corresponding to Weather Conditions for PV Generation Systems". *IEEE Transaction on Industrial Electronics Vol 53 No 4 August 2006.*

N. S. D'Souza, L. A. C. Lopes, and X. Liu, "An intelligent maximum power point tracker using peak current control," in *Proc. 36th Annu. IEEE Power Electron. Spec. Conf.*, 2005, pp. 172–177.

Noguchi Toshihiko, Togashi Shigeroni, "Short-current Pulse Based Adaptive Maximum Power Point Tracking for the Photovoltaic Power Generation System", *Industrial Electronics, 2000. ISIE 2000. Proceedings of the 2000 IEEE International Symposium on Volume 1, 4-8 Dec. 2000 Page(s):157 - 162 vol.1.*

O. Wasynczuk, “Dynamic behavior of a class of photovoltaic power systems,”*IEEE Trans. Power App. Syst.*, vol. 102, no. 9, pp. 3031–3037, Sep. 1983.

S. Yuvarajan and S. Xu, “Photo-voltaic power converter with a simple maximum-power-point-tracker,” in *Proc. 2003 Int. Symp. Circuits Syst.*, 2003, pp. III-399–III-402.

Trishan Esram and Patrick L. Chapman, "Comparison of photovoltaic array maximum power point tracking techniques ", *IEEE Transactions on energy conversion, vol. 36, Issue 3, pp. 1016-1019, July 2006.*

T. Hiyama, S. Kouzuma, and T. Imakubo, “Identification of optimal operating point of PV modules using neural network for real time maximum power tracking control,” *IEEE Trans. Energy Convers.*, vol. 10, no. 2, pp. 360–367, Jun. 1995.

T. Noguchi, S. Togashi, and R. Nakamoto, “Short-current pulse based adaptive maximum-power-point tracking for photovoltaic power generation system,” in *Proc. 2000 IEEE Int. Symp. Ind. Electron.*, 2000, pp. 157–162.

T. Senju and K. Uezato. "Maximum power point tracker using fuzzy control for photovoltaic arrays", in Proc. *IEEE Int. Conf Industrial Technology,* pp. 143-147, 1994

T. Tafticht and K. Agbossou, “Development of a MPPT method for photovoltaic systems,” in *Canadian Conf. Elect. Comput. Eng.*, 2004, pp. 1123– 1126.

Victor Marrero1, Jose Colom, Jessenia Merced, Solimar Reyes, Diego Arias1, Lionel Orama, Wilson Rivera, and Brian Donovan, “Work in Progress - Multi-Campus and Multi-Disciplinary Student-Led Testbed at UPRM”, *35th ASEE/IEEE Frontiers in Education Conference October 19 – 22, 2005, Indianapolis, IN.*

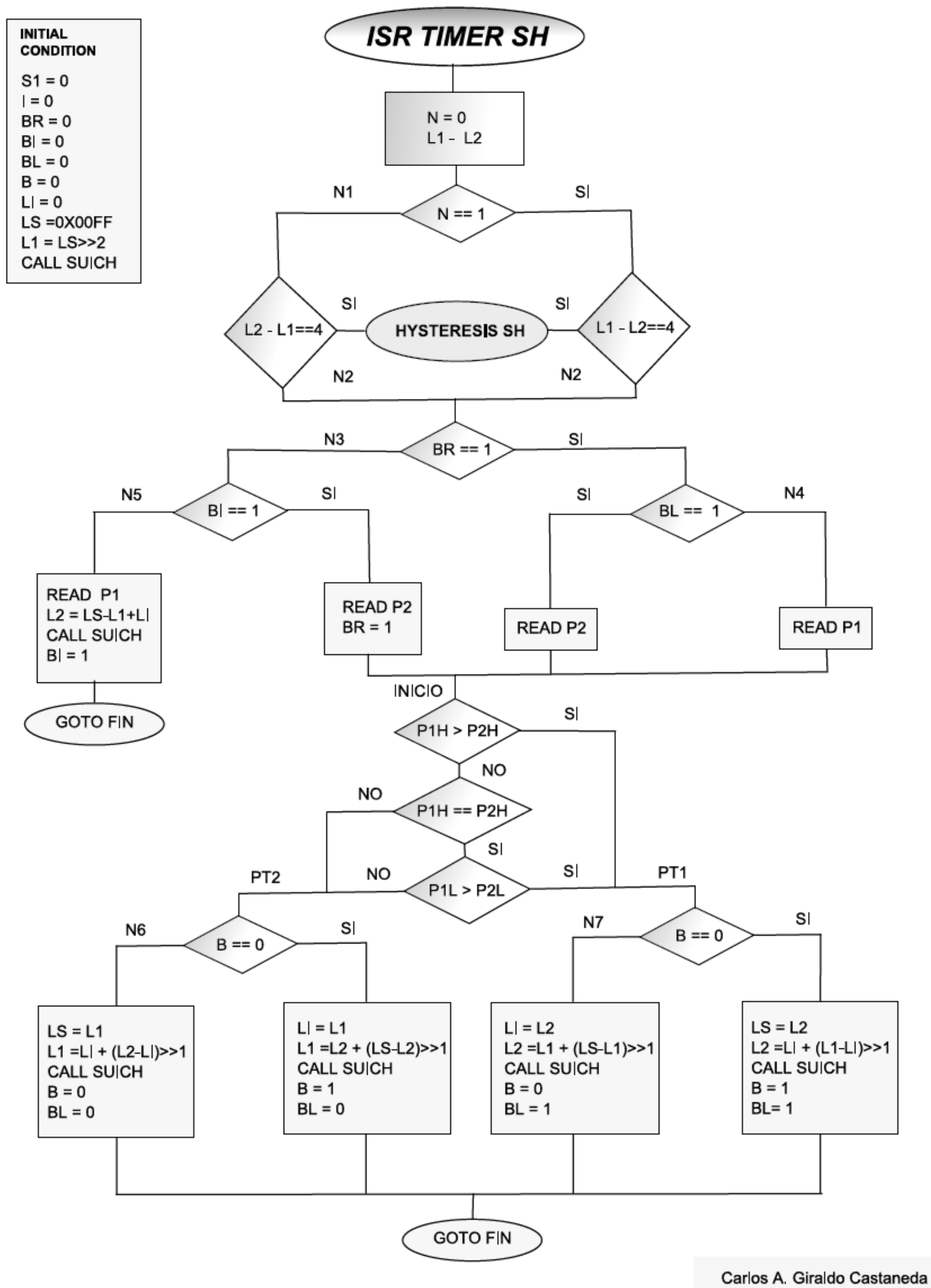
Youngseok Jung, Junghun So, Gwonjong Yu, Jaeho Choi, “Improved perturbation and observation method (IP&O) of MPPT control for photovoltaic power systems”. *Photovoltaic Specialists Conference, 2005. Conference Record of the Thirty-first IEEE 3-7 Jan. 2005 Page(s):1788 – 1791.*

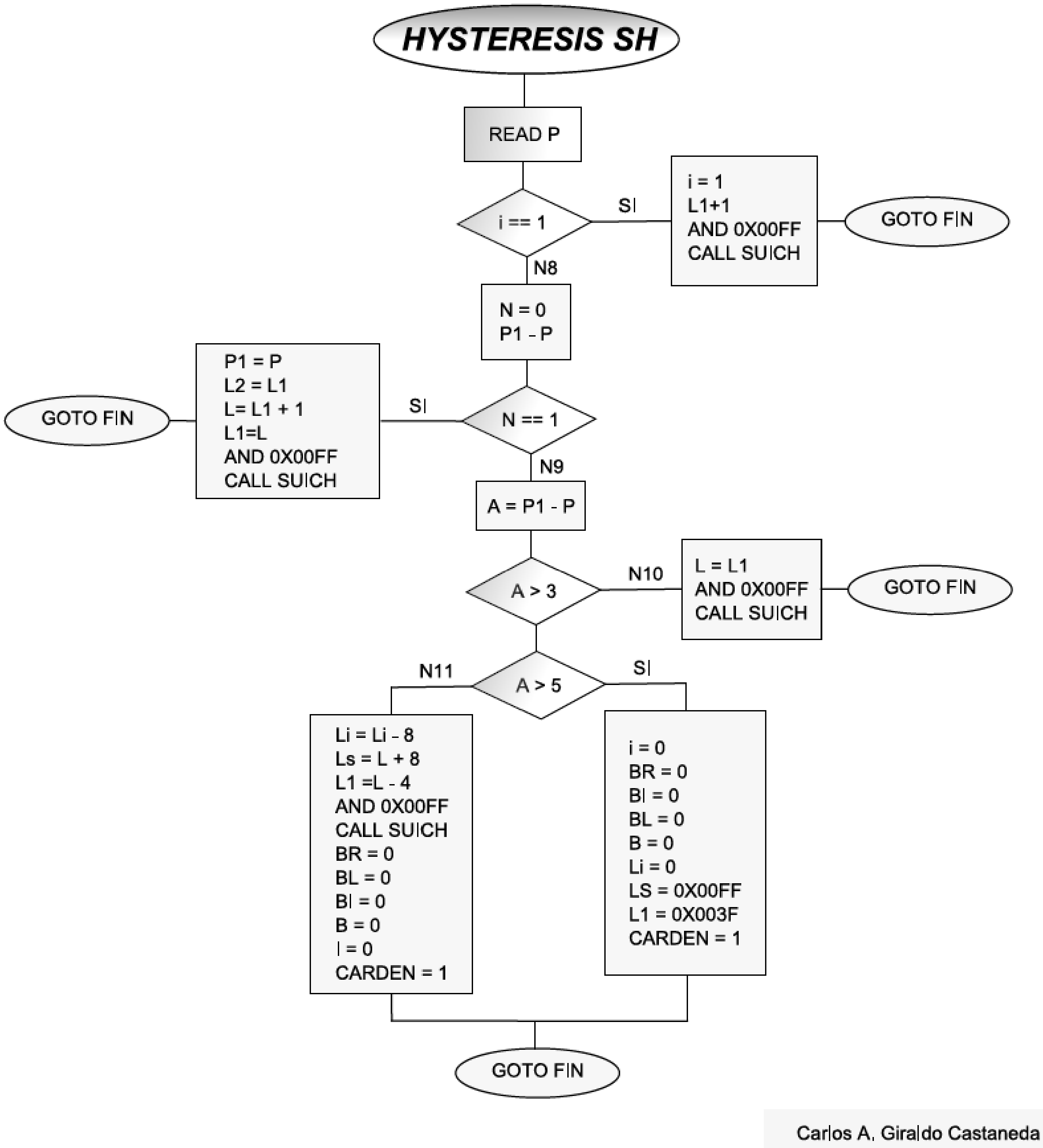
Ying-Tung Hsiao; China-HongChen; “Maximum power tracking for photovoltaic power system”, *Industry Applications Conference, 2002. 37th IAS Annual Meeting. Conference Record of the Volume 2, 13-18 Oct. 2002 Page(s):1035 - 1040 vol.2.*

APPENDIX A. FIRMWARE

The following firmware carried out all of the control functions of the embedded system, including the communication protocols between the PC and the A/D converters network, for the maximum power point tracking systems.

APPENDIX A1 FIRWARE SELECTIVE HOPPING





HYSTERESIS SH
READ P
i == 1
SI
i = 1
L1+1
AND 0X00FF
CALL SUICH
GOTO FIN
N8
N = 0
P1 - P
P1 = P
L2 = L1
L= L1 + 1
L1=L
AND 0X00FF
CALL SUICH
GOTO FIN
SI
N == 1
N9
A = P1 - P
A > 3
N10
L = L1
AND 0X00FF
CALL SUICH
GOTO FIN
N11
A > 5
SI
Li = Li - 8
Ls = L + 8
L1 =L - 4
AND 0X00FF
CALL SUICH
BR = 0
BL = 0
BI = 0
B = 0
I = 0
CARDEN = 1
i = 0
BR = 0
BI = 0
BL = 0
B = 0
Li = 0
LS = 0X00FF
L1 = 0X003F
CARDEN = 1
GOTO FIN
Carlos A. Giraldo Castaneda

```
;==============================================================================
		.equ __30F4013, 1
		.include "p30f4013.inc"
;---------------------------------------------------------------------
;Global Declarations
		.global __reset ;The label for the first line of code
		.global __OscillatorFail ;Declare Oscillator Fail trap routine label
		.global __AddressError ;Declare Address Error trap routine label
		.global __StackError ;Declare Stack Error trap routine label
		.global __MathError ;Declare Math Error trap routine label
		.global __U2RXInterrupt;Declare UART 2 RX ISR label
		.global __T1Interrupt;

;---------------------------------------------------------------------
;RAM variables
.bss
;SELECT: .space 2 ;Allocating space (in bytes) to variable

RXAUX: .space 2 ;Allocating space (in bytes) to variable
FLAGS: .space 2 ;Allocating space (in bytes) to variable
BYTESAV: .space 2 ;Allocating space (in bytes) to variable
CONTX: .space 2 ;Allocating space (in bytes) to variable
ACBYTE: .space 2 ;Allocating space (in bytes) to variable
REGIS1: .space 2 ;Allocating space (in bytes) to variable
REGIS2: .space 2 ;Allocating space (in bytes) to variable
VOLTAJE: .space 2 ;Allocating space (in bytes) to variable
CORRIENTE: .space 2 ;Allocating space (in bytes) to variable
L1: .space 2 ;Allocating space (in bytes) to variable
L2: .space 2 ;Allocating space (in bytes) to variable
P1L: .space 2 ;Allocating space (in bytes) to variable
P1H: .space 2 ;Allocating space (in bytes) to variable
P2L: .space 2 ;Allocating space (in bytes) to variable
P2H: .space 2 ;Allocating space (in bytes) to variable
PH: .space 2 ;Allocating space (in bytes) to variable
PL: .space 2 ;Allocating space (in bytes) to variable
LI: .space 2 ;Allocating space (in bytes) to variable
LS: .space 2 ;Allocating space (in bytes) to variable
L: .space 2 ;Allocating space (in bytes) to variable

.section MyDataSection, bss, near
BUFFER: .space 1024 ;Allocating space (in bytes) to variable
;---------------------------------------------------------------------
;BIT Definitions
		.equiv CARDEN,	0x0000
		.equiv BFULL,	0x0001
		.equiv BR,			0x0002
		.equiv BI,		0x0003
		.equiv BL,		0x0004
		.equiv B,		0x0005
		.equiv I,		0x0006
		.equiv IS,		0x0007
;-----------------------------------------------------------------------------
;Configuration bits

   config __FOSC, CSW_FSCM_OFF & ECIO_PLL4
		config __FWDT, WDT_OFF
   config __FBORPOR, PBOR_ON & BORV_27 & PWRT_64 & MCLR_EN
		config __FGS, CODE_PROT_OFF
;-----------------------------------------------------------------------------
;Program Specific Constants (literals used in code)
;			.equ FCY, #7372800 ;Instruction cycle rate (Osc x PLL / 4)
;==============================================================================
;Start of code
			.text ;Start of Code section
```

```
;-----------------------------------------------------------------------------
;Initialize stack pointer and limit register
__reset:     mov #__SP_init, W15 ;Initialize the Stack Pointer register
                    mov #__SPLIM_init, W0 ;Get address at the end of stack space
                    mov W0, SPLIM ;Load the Stack Pointer Limit register
                    nop ;Add NOP to follow SPLIM initialization

;-----------------------------------------------------------------------------
;INICIALIZACION PUERTOS

            CLR LATB                ;Initialize to LOW all output pins
            CLR LATA                ;Initialize to LOW all output pins
            CLR LATD                ;Initialize to LOW all output pins
            CLR LATF                ;Initialize to LOW all output pins
            CLR TRISA               ;
            MOV #0x0007, W0 ;
            MOV W0, TRISB       ;RB12:RB3->SALIDAS Y RB2:RB0->ENTRADAS
            CLR TRISD               ;
            MOV #0x0017, W0 ;
            MOV W0, TRISF       ;RB12:RB3->SALIDAS Y RB2:RB0->ENTRADAS
            ;BSET INTCON1,#NSTDIS       ; NESTING DISABLE


;-----------------------------------------------------------------------------
;INICIALIZACION TIMER1

            MOV #0x0030,W0          ;
            MOV W0,T1CON            ; PRESC 1:256 Y 16 BIT MODE
            CLR TMR1                        ;
            MOV #0x1200,W0          ; PERIODO = 300ms
            MOV W0,PR1                      ; PR2 = PERIODO*(FCY=7372800)/256
            BSET IPC0,#T1IP0    ; PRIORIDAD=7
            BSET IPC0,#T1IP1    ;
            BSET IPC0,#T1IP2    ;
            BCLR IFS0,#T1IF         ; BORRA EL FLAG T2IF
            BSET IEC0,#T1IE         ; HABILITA LA INTERRUPCION
            BSET T1CON,#TON         ; HABILITA EL TIMER
;-----------------------------------------------------------------------------
;Initialize UART

            MOV #0x0003,W0          ;
            MOV W0,U2BRG            ; Configure Baud Rate equal to 115200 bps
            CLR U2MODE                      ; sets 8 data bits, 1 stop bit, no parity
            MOV #0x0110,W0          ;
            MOV W0,U2STA            ; Enables RX interrupt when a character is received
            BSET U2MODE,#UARTEN     ;
            BSET U2STA,#UTXEN       ;
            BCLR IFS1,#U2TXIF       ;
            BCLR IFS1,#U2RXIF       ;
            BCLR IPC6,#U2RXIP0      ; PRIORIDAD=6
            BSET IPC6,#U2RXIP1      ;
            BSET IPC6,#U2RXIP2      ;
            BSET IEC1,#U2RXIE       ; Enables UART 2 RX interrupt
;-----------------------------------------------------------------------------
;INICIALIZACION ADC

            MOV #0xFFF8,W0          ;RB2 ANALOG
            MOV W0,ADPCFG           ;
            MOV #0x00E4,W0          ;
            MOV W0,ADCON1           ; AUTOSAMPLING, INTEGER FORMAT, INTERNAL SAMP COUNTER
            MOV #0x0006,W0          ;
            MOV W0,ADCSSL           ; MUESTREA CANALES 1 Y 2
            MOV #0x1F1D,W0          ; TAD=(ADCS=29+1)/2=2.034us, SAMC=31TAD
            MOV W0,ADCON3           ; CLOCK SOURCE = SYSTEM CLOCK
            MOV #0x2408,W0          ; VREFH=VREF+ PIN, VREFL=AVSS, NO ESCANEA;
```

```
                MOV W0,ADCON2           ; INTERRUPCION CADA 2 CONVERSIONES Y ALTERNA ENTRE MUXA Y
                                        MUX B
                BSET ADCON1,#ADON       ; HABILITA EL MODULO AD

;------------------------------------------------------------------------------
;Initialize Registers

                CLR BYTESAV                     ;
                CLR CONTX                       ;
                CLR FLAGS                       ;
                CLR LI                          ;
                MOV #0x00FF,W0          ;
                MOV W0,LS                       ;
                CLR W0                          ;
                CALL SUICH                      ;
;------------------------------------------------------------------------------
MAINLOOP:
                MOV #0x0040,W0          ; W0=64
                XOR BYTESAV,WREG        ; BYTES DISPONIBLES=512?
                BRA Z,BUFF1                     ;

                BTSC FLAGS,#BFULL       ; BUFFER FULL?
                GOTO BUFF2                      ;
                GOTO MAINLOOP           ;

BUFF1:          BTSS FLAGS,#CARDEN      ; TARGETA HABILITADA?
                GOTO MAINLOOP           ; NO
                CLR CONTX                       ;
                MOV #BUFFER,W4          ;
VUEL1:          MOV.B [W4++],W0         ;
                MOV.B WREG,U2TXREG      ; TRANSMITE BYTE
T1:             BTSS IFS1,#U2TXIF ; PRUEBA HASTA TRANSMITIR TODO
                GOTO T1                 ;
                BCLR IFS1,#U2TXIF       ; BORRA FLAG
                INC CONTX                       ; INCREMENTA CONTEO DE BYTES TRANSMITIDOS
                MOV #0x0040,W0          ; W0=512
                XOR CONTX,WREG          ; BYTES TRANSMITIDOS=512?
                BRA NZ,VUEL1            ; NO->CONTINUE TRANSMITIENDO
                GOTO MAINLOOP           ; SI->FINALICE
BUFF2:          BCLR FLAGS,#BFULL       ;
                BTSS FLAGS,#CARDEN      ; TARGETA HABILITADA?
                GOTO MAINLOOP           ; NO
                CLR CONTX                       ;
                MOV #BUFFER,W4          ;
                ADD #0x0040,W4          ; W4=#BUFFER+512;
VUEL2:          MOV.B [W4++],W0         ;
                MOV.B WREG,U2TXREG      ; TRANSMITE BYTE
T2:             BTSS IFS1,#U2TXIF ; PRUEBA HASTA TRANSMITIR TODO
                GOTO T2                 ;
                BCLR IFS1,#U2TXIF       ; BORRA FLAG
                INC CONTX                       ; INCREMENTA CONTEO DE BYTES TRANSMITIDOS
                MOV #0x0040,W0          ; W0=512
                XOR CONTX,WREG          ; BYTES TRANSMITIDOS=512?
                BRA NZ,VUEL2            ; NO->CONTINUE TRANSMITIENDO
                GOTO MAINLOOP           ; SI->FINALICE
;==============================================================================
__OscillatorFail:
                        NOP                             ;
                        bra __OscillatorFail ;Loop forever when oscillator failure occurs
;------------------------------------------------------------------------------
;Address Error trap routine
__AddressError:
                        NOP                             ;
                        bra __AddressError ;Loop forever when address error occurs
;------------------------------------------------------------------------------
```

```
;Stack Error trap routine
__StackError:
                    NOP                         ;
                    bra __StackError ;Loop forever when stack error occurs
;-----------------------------------------------------------------------------
;Math (Arithmetic) Error trap routine
__MathError:
                    NOP                         ;
                    bra __MathError ;Loop forever when math error occurs
;-----------------------------------------------------------------------------
;Math (Arithmetic) Error trap routine
__U2RXInterrupt:
                    PUSH W0                             ;
                    BCLR IFS1,#U2RXIF           ; BORRA FLAG DE RECEPCION UART

                    MOV U2RXREG,W0              ;
                    AND #0x00FF,W0              ; BORRA LOS 8 MSBS DE U2RXREG
                    MOV W0,RXAUX                ; ALMACENA U2RXREG EN RXAUX
                    MOV #0x00AA,W1              ;
                    CPSEQ W0,W1                         ;
                    GOTO P1                             ;
                    BSET FLAGS,#CARDEN          ;
                    GOTO P2                             ;
            P1:     MOV #0x0055,W1              ;
                    CPSEQ W0,W1                         ;
                    GOTO P2
                    BCLR FLAGS,#CARDEN          ;
            P2:     POP W0                              ;
                    RETFIE
;-----------------------------------------------------------------------------
;T1 Interrupt routine
__T1Interrupt:
            PUSH W0                             ;
            BCLR IFS0,#T1IF             ;
    AD:     BTSS ADCON1,#DONE           ;
            GOTO AD                             ;
            MOV #BUFFER,W1              ;
            MOV BYTESAV,W2              ;
            MOV ADCBUF1,WREG            ;
            MOV WREG, VOLTAJE           ;
            MOV W0,[W1+W2]              ;
            INC2 BYTESAV                ;
            MOV BYTESAV,W2              ;
            MOV ADCBUF2,WREG            ;
            MOV WREG,CORRIENTE          ;
            MOV W0,[W1+W2]              ;
            INC2 BYTESAV                ;
            MOV #0x0080,W0              ;
            CP BYTESAV                          ;
            BRA NZ,F2                           ;
            CLR BYTESAV                         ;
            BSET FLAGS,#BFULL           ;

    F2:     BTSS FLAGS,#CARDEN  ;
            GOTO REST1                  ;

            BTSC FLAGS,#IS              ;
            GOTO IN                             ;
            MOV #0x0020,W0              ; SUICHEO INICIAL
            MOV W0,L1                           ;
            CALL SUICH                          ;
            BSET FLAGS,#IS              ;
            GOTO FIN                            ;

            ; MPPT SH
```

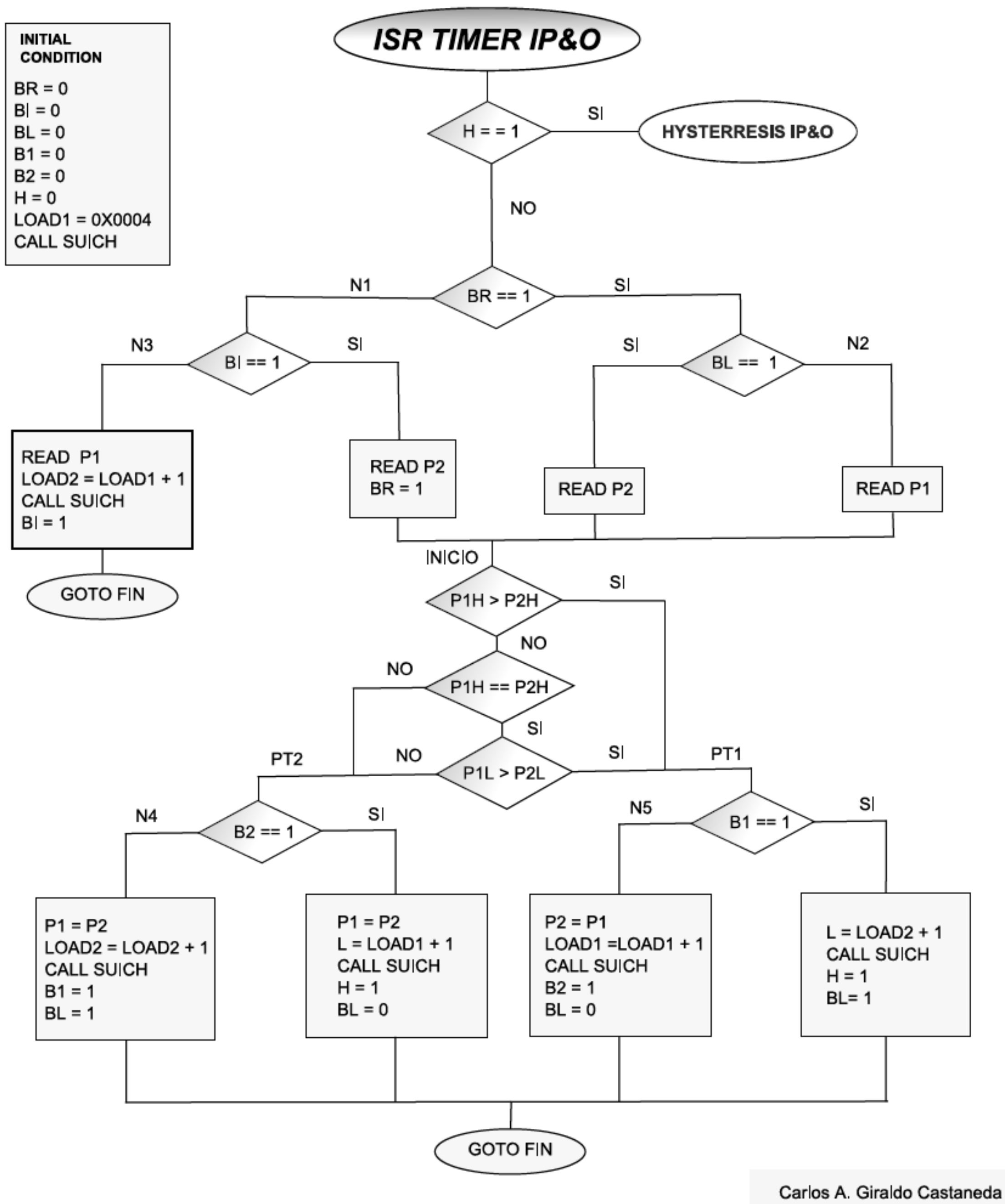
```
IN:     BCLR SR,#N            ;
        MOV L1,W0             ;
        MOV L2,W1             ;
        SUB W0,W1,W2          ;
        BTSS SR,#N            ;
        GOTO N1               ;
        SUB W1,W0,W2          ;
        MOV #0x0001,W1        ;
        CPSEQ W2,W1           ;
        GOTO N2               ;
        GOTO HISTERESIS       ;
N1:     MOV #0x0001,W1        ;
        CPSEQ W2,W1           ;
        GOTO N2               ;
        GOTO HISTERESIS       ;

N2:     BTSS FLAGS,#BR        ;
        GOTO N3               ;
        BTSS FLAGS,#BL        ;
        GOTO N4               ;
        MOV VOLTAJE,W0        ;
        MUL.W CORRIENTE       ;
        MOV W2,P2L            ;
        MOV W3,P2H            ;
        GOTO INICIO           ;
N4:     MOV VOLTAJE,W0        ;
        MUL.W CORRIENTE       ;
        MOV W2,P1L            ;
        MOV W3,P1H            ;
        GOTO INICIO           ;
N3:     BTSS FLAGS,#BI        ;
        GOTO N5               ;
        MOV VOLTAJE,W0        ;
        MUL.W CORRIENTE       ;
        MOV W2,P2L            ;
        MOV W3,P2H            ;
        BSET FLAGS,#BR        ;
        GOTO INICIO           ;
N5:     MOV VOLTAJE,W0        ;
        MUL.W CORRIENTE       ;
        MOV W2,P1L            ;
        MOV W3,P1H            ;
        MOV LS,W1             ;
        MOV L1,W2             ;
        SUB W1,W2,W3          ;
        MOV LI,W1             ;
        ADD W1,W3,W0          ;
        AND #0x00FF,W0        ;
        MOV W0,L2             ;
        CALL SUICH            ;
        BSET FLAGS,#BI        ;
        GOTO FIN              ;
INICIO: MOV P1H,W0            ;
        MOV P2H,W1            ;
        CPSGT W1,W0           ;
        GOTO PT1              ;
        CPSEQ W0,W1           ;
        GOTO PT2              ;
        MOV P1L,W0            ;
        MOV P2L,W1            ;
        CPSGT W1,W0           ;
        GOTO PT1              ;
PT2:    BTSC FLAGS,#B         ;
        GOTO N6               ;
        MOV L1,W0             ;
```

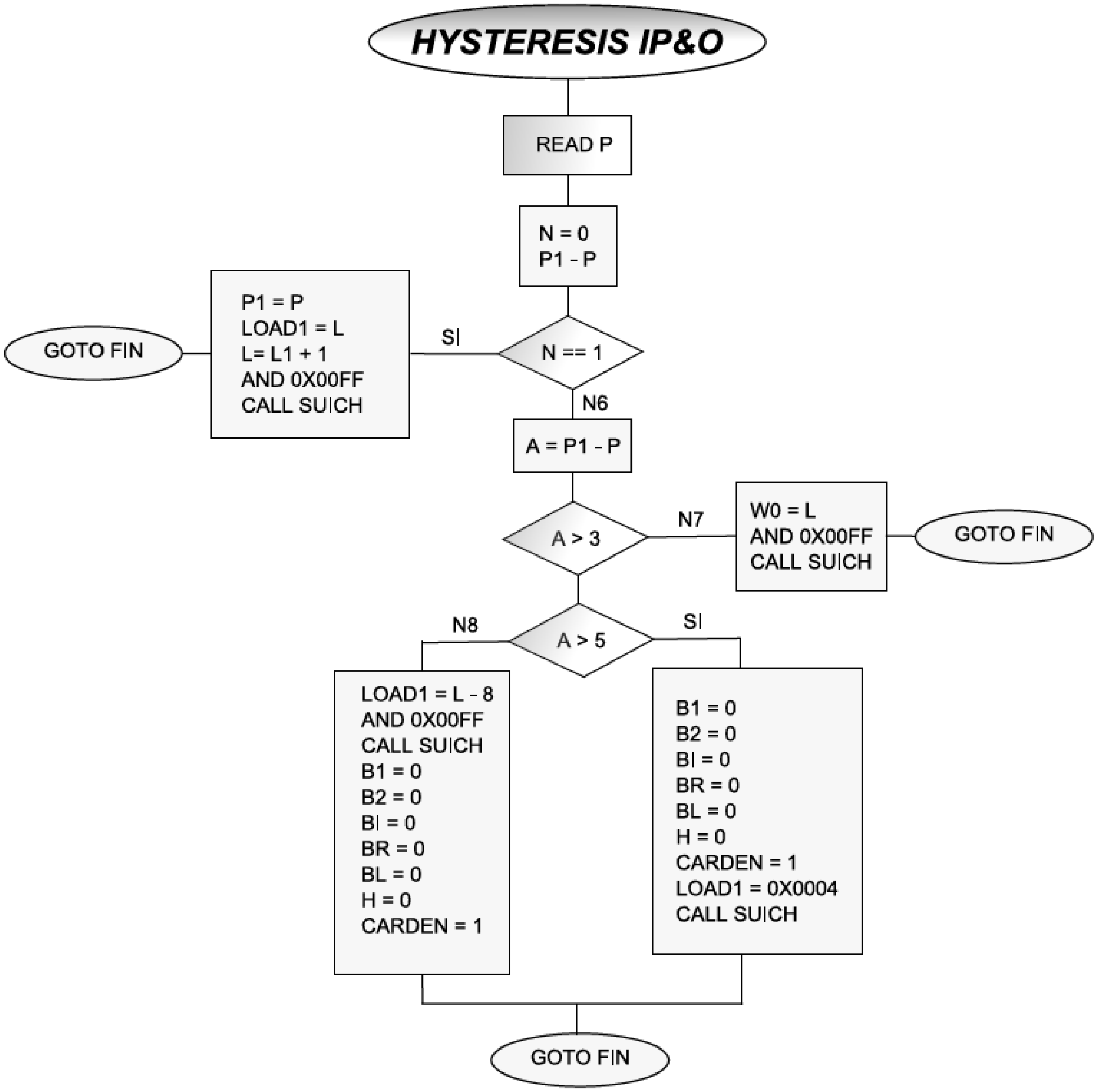
```
                MOV W0,LI                       ;
                MOV LS,W0                       ;
                MOV L2,W1                       ;
                SUB W0,W1,W2            ;
                LSR W2,W4                       ;
                ADD W1,W4,W0            ;
                AND #0x00FF,W0          ;
                MOV W0,L1                       ;
                CALL SUICH                      ;
                BSET FLAGS,#B           ;
                BCLR FLAGS,#BL          ;
                GOTO FIN                        ;
        N6:     MOV L1,W0                       ;
                MOV W0,LS                       ;
                MOV L2,W0                       ;
                MOV LI,W1                       ;
                SUB W0,W1,W2    ;
                LSR W2,W4                       ;
                ADD W1,W4,W0            ;
                AND #0x00FF,W0          ;
                MOV W0,L1                       ;
                CALL SUICH                      ;
                BCLR FLAGS,#B           ;
                BCLR FLAGS,#BL          ;
                GOTO FIN                        ;
   PT1:         BTSC FLAGS,#B           ;
                GOTO N7                         ;
                MOV L2,W0                       ;
                MOV W0,LS                       ;
                MOV L1,W0                       ;
                MOV LI,W1                       ;
                SUB W0,W1,W2            ;
                LSR W2,W4                       ;
                ADD W1,W4,W0            ;
                AND #0x00FF,W0          ;
                MOV W0,L2                       ;
                CALL SUICH                      ;
                BSET FLAGS,#B           ;
                BSET FLAGS,#BL          ;
                GOTO FIN                        ;
        N7:     MOV L2,W0                       ;
                MOV W0,LI                       ;
                MOV LS,W0                       ;
                MOV L1,W1                       ;
                SUB W0,W1,W2            ;
                LSR W2,W4                       ;
                ADD W1,W4,W0            ;
                AND #0x00FF,W0          ;
                MOV W0,L2                       ;
                CALL SUICH                      ;
                BCLR FLAGS,#B           ;
                BSET FLAGS,#BL          ;
                GOTO FIN                        ;
;----------------------------------------------------------------------------
HISTERESIS:             MOV VOLTAJE,W0          ;
                        MUL.W CORRIENTE                 ;
                        MOV #0x04B4,W1          ;
                        CPSGT W0,W1                     ;
                        GOTO REST                       ;
                        MOV W2,PL                       ;
                        MOV W3,PH                       ;
                        BTSC FLAGS,#I           ;
                        GOTO N8                         ;
                        BSET FLAGS,#I           ;
                        MOV L1,W0                       ;
```

```
                ADD #0x0001,W0              ;
                AND #0x00FF,W0              ;
                CALL SUICH                              ;
                GOTO FIN                                ;

        N8:     MOV P1H,W4                              ;
                BCLR SR,#N                              ;
                SUB W4,W3,W1                ;
                BTSS SR,#N                              ;
                GOTO N9                                 ;
                MOV PH,W0                               ;
                MOV PL,W1                               ;
                MOV W0,P1H                              ;
                MOV W1,P1L                              ;
                MOV L1,W0                               ;
                MOV W0,L2                               ;
                ADD #0x0001,W0              ;
                AND #0x00FF,W0              ;
                MOV W0,L                                ;
                MOV W0,L1                               ;
                CALL SUICH                              ;
                GOTO FIN                                ;
        N9:     MOV #0x0004,W0              ;
                CPSGT W1,W0                             ;
                GOTO N10                                ;
                MOV #0x0007,W0              ;
                CPSGT W1,W0                             ;
                GOTO N11                                ;
                GOTO REST                               ;
N10:             MOV L1,W0                              ;
                AND #0x00FF,W0              ;
                MOV W0,L                                ;
                CALL SUICH                              ;
                GOTO FIN                                ;
N11:            GOTO REST
                ;MOV L,W0                               ;
                ;SUB #0x0004,W0             ;
                ;AND #0x00FF,W0             ;
                ;MOV W0,L1                              ;
                ;CALL SUICH                             ;
                ;MOV L,W0                               ;
                ;ADD #0x0008,W0             ;
                ;AND #0x00FF,W0             ;
                ;MOV W0,LS                              ;
        ;       MOV LI,W0                               ;
        ;       BCLR SR,#N                              ;
        ;       SUB #0x0008,W0              ;
        ;       BTSC SR,#N                              ;
        ;       GOTO REST                               ;
        ;       MOV W0,LI                               ;
        ;       CLR FLAGS                               ;
        ;       BSET FLAGS,#CARDEN          ;
        ;       GOTO FIN                                ;
REST:           CLR FLAGS                               ;
                BSET FLAGS,#CARDEN          ;
                CLR LI                                  ;
                MOV #0x00FF,W0              ;
                MOV W0,LS                               ;
                ;MOV #0x003F,W0             ;
                ;MOV W0,L1                              ;
                ;CALL SUICH                             ;
                GOTO FIN
REST1:          CLR FLAGS                               ;
                CLR LI                                  ;
                MOV #0x00FF,W0              ;
```

```
                    MOV W0,LS                       ;
                    CLR W0                          ;
                    CALL SUICH                      ;

        FIN:    POP W0                          ;
                RETFIE                          ;
;=============================================================================
SUICH:          MOV W0, ACBYTE
                MOV ACBYTE,WREG                 ; PALABRA 2-BYTES
                MOV WREG,REGIS1                 ;
                RLNC REGIS1                     ;
                RLNC REGIS1                     ;
                RLNC REGIS1,WREG        ;
                MOV WREG,LATB           ;
                MOV ACBYTE,WREG                 ;
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                MOV W0,REGIS1           ;
                MOV #0x000C,W0          ;
                AND REGIS1                      ;
                MOV ACBYTE,WREG                 ;
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                MOV W0,REGIS2           ;
                MOV #0x0300,W0          ;
                AND REGIS2,WREG         ;
                IOR REGIS1,WREG         ;
                MOV WREG,LATD           ;
                BTSS ACBYTE,#15         ;
                GOTO SALTO1                     ;
                BSET LATA,#11           ;
                GOTO SALTO2                     ;
SALTO1:         BCLR LATA,#11           ;
SALTO2:         RETURN                          ;

        .end ;End of code in this file
```

APPENDIX A2 FIRWARE IMPROVED PERTURBATION AND OBSERVATION





Carlos A. Giraldo Castaneda

```
;==============================================================================
        .equ __30F4013, 1
        .include "p30f4013.inc"
;----------------------------------------------------------------------
;Global Declarations
        .global __reset ;The label for the first line of code
        .global __OscillatorFail ;Declare Oscillator Fail trap routine label
        .global __AddressError ;Declare Address Error trap routine label
        .global __StackError ;Declare Stack Error trap routine label
        .global __MathError ;Declare Math Error trap routine label
        .global __U2RXInterrupt;Declare UART 2 RX ISR label
        .global __T1Interrupt;

;----------------------------------------------------------------------
;RAM variables
.bss
;SELECT: .space 2 ;Allocating space (in bytes) to variable
RXAUX: .space 2 ;Allocating space (in bytes) to variable
FLAGS: .space 2 ;Allocating space (in bytes) to variable
BYTESAV: .space 2 ;Allocating space (in bytes) to variable
CONTX: .space 2 ;Allocating space (in bytes) to variable
ACBYTE: .space 2 ;Allocating space (in bytes) to variable
REGIS1: .space 2 ;Allocating space (in bytes) to variable
REGIS2: .space 2 ;Allocating space (in bytes) to variable
VOLTAJE: .space 2 ;Allocating space (in bytes) to variable
CORRIENTE: .space 2 ;Allocating space (in bytes) to variable
LOAD1: .space 2 ;Allocating space (in bytes) to variable
LOAD2: .space 2 ;Allocating space (in bytes) to variable
P1L: .space 2 ;Allocating space (in bytes) to variable
P1H: .space 2 ;Allocating space (in bytes) to variable
P2L: .space 2 ;Allocating space (in bytes) to variable
P2H: .space 2 ;Allocating space (in bytes) to variable
PL: .space 2 ;Allocating space (in bytes) to variable
PH: .space 2 ;Allocating space (in bytes) to variable
L: .space 2 ;Allocating space (in bytes) to variable

.section MyDataSection, bss, near
BUFFER: .space 1024 ;Allocating space (in bytes) to variable
;----------------------------------------------------------------------
;BIT Definitions
        .equiv CARDEN,      0x0000
        .equiv BFULL,       0x0001
        .equiv B1,          0x0002
        .equiv B2,          0x0003
        .equiv BI,          0x0004
        .equiv BR,                      0x0005
        .equiv BL,          0x0006
        .equiv H,           0x0007
        .equiv IS,          0x0008
;------------------------------------------------------------------------------
;Configuration bits

   config __FOSC, CSW_FSCM_OFF & ECIO_PLL4
        config __FWDT, WDT_OFF
   config __FBORPOR, PBOR_ON & BORV_27 & PWRT_64 & MCLR_EN
        config __FGS, CODE_PROT_OFF

;------------------------------------------------------------------------------
;Program Specific Constants (literals used in code)
;               .equ FCY, #7372800 ;Instruction cycle rate (Osc x PLL / 4)
;==============================================================================
;Start of code

                .text ;Start of Code section
;------------------------------------------------------------------------------
```

```
;Initialize stack pointer and limit register
__reset:	mov #__SP_init, W15 ;Initialize the Stack Pointer register
			mov #__SPLIM_init, W0 ;Get address at the end of stack space
			mov W0, SPLIM ;Load the Stack Pointer Limit register
			nop ;Add NOP to follow SPLIM initialization
;-----------------------------------------------------------------------------
;INICIALIZACION PUERTOS

		CLR LATB			;Initialize to LOW all output pins
		CLR LATA			;Initialize to LOW all output pins
		CLR LATD			;Initialize to LOW all output pins
		CLR LATF			;Initialize to LOW all output pins

		CLR TRISA			;
		MOV #0x0007, W0 ;
		MOV W0, TRISB			;RB12:RB3->SALIDAS Y RB2:RB0->ENTRADAS
		CLR TRISD			;
		MOV #0x0017, W0 ;
		MOV W0, TRISF			;RB12:RB3->SALIDAS Y RB2:RB0->ENTRADAS

		;BSET INTCON1,#NSTDIS		; NESTING DISABLE
;-----------------------------------------------------------------------------
;INICIALIZACION TIMER1

		MOV #0x0030,W0			;
		MOV W0,T1CON			; PRESC 1:256 Y 16 BIT MODE
		CLR TMR1				;
		MOV #0x1200,W0			; PERIODO = 300ms
		MOV W0,PR1				; PR2 = PERIODO*(FCY=7372800)/256
		BSET IPC0,#T1IP0		; PRIORIDAD=7
		BSET IPC0,#T1IP1		;
		BSET IPC0,#T1IP2		;
		BCLR IFS0,#T1IF			; BORRA EL FLAG T2IF
		BSET IEC0,#T1IE			; HABILITA LA INTERRUPCION
		BSET T1CON,#TON			; HABILITA EL TIMER
;-----------------------------------------------------------------------------
;Initialize UART

		MOV #0x0003,W0			;
		MOV W0,U2BRG			; Configure Baud Rate equal to 115200 bps
		CLR U2MODE				; sets 8 data bits, 1 stop bit, no parity
		MOV #0x0110,W0			;
		MOV W0,U2STA			; Enables RX interrupt when a character is received
		BSET U2MODE,#UARTEN		;
		BSET U2STA,#UTXEN		;
		BCLR IFS1,#U2TXIF		;
		BCLR IFS1,#U2RXIF		;
		BCLR IPC6,#U2RXIP0		; PRIORIDAD=6
		BSET IPC6,#U2RXIP1		;
		BSET IPC6,#U2RXIP2		;
		BSET IEC1,#U2RXIE		; Enables UART 2 RX interrupt
;-----------------------------------------------------------------------------
;INICIALIZACION ADC

		MOV #0xFFF8,W0			;RB2 ANALOG
		MOV W0,ADPCFG			;
		MOV #0x00E4,W0			;
		MOV W0,ADCON1			; AUTOSAMPLING, INTEGER FORMAT, INTERNAL SAMP COUNTER
		MOV #0x0006,W0			;
		MOV W0,ADCSSL			; MUESTREA CANALES 1 Y 2
		MOV #0x1F1D,W0			; TAD=(ADCS=29+1)/2=2.034us, SAMC=31TAD
		MOV W0,ADCON3			; CLOCK SOURCE = SYSTEM CLOCK
		MOV #0x2408,W0			; VREFH=VREF+ PIN, VREFL=AVSS, NO ESCANEA;
		MOV W0,ADCON2			; INTERRUPCION CADA 2 CONVERSIONES Y ALTERNA ENTRE MUXA Y
						MUX B
```

```
                BSET ADCON1,#ADON           ; HABILITA EL MODULO AD

;------------------------------------------------------------------------------
;Initialize Registers
                CLR BYTESAV                         ;
                CLR CONTX                           ;
                CLR FLAGS                           ;
                CLR W0                              ;
                CALL SUICH                          ;
;------------------------------------------------------------------------------
MAINLOOP:
                MOV #0x0040,W0              ; W0=64
                XOR BYTESAV,WREG            ; BYTES DISPONIBLES=512?
                BRA Z,BUFF1                         ;
                BTSC FLAGS,#BFULL           ; BUFFER FULL?
                GOTO BUFF2                          ;
                GOTO MAINLOOP               ;

BUFF1:          BTSS FLAGS,#CARDEN          ; TARGETA HABILITADA?
                GOTO MAINLOOP               ; NO
                CLR CONTX                           ;
                MOV #BUFFER,W4              ;
VUEL1:          MOV.B [W4++],W0             ;
                MOV.B WREG,U2TXREG          ; TRANSMITE BYTE
T1:             BTSS IFS1,#U2TXIF ; PRUEBA HASTA TRANSMITIR TODO
                GOTO T1                     ;
                BCLR IFS1,#U2TXIF           ; BORRA FLAG
                INC CONTX                           ; INCREMENTA CONTEO DE BYTES TRANSMITIDOS
                MOV #0x0040,W0              ; W0=512
                XOR CONTX,WREG              ; BYTES TRANSMITIDOS=512?
                BRA NZ,VUEL1                ; NO->CONTINUE TRANSMITIENDO
                GOTO MAINLOOP               ; SI->FINALICE
BUFF2:          BCLR FLAGS,#BFULL           ;
                BTSS FLAGS,#CARDEN          ; TARGETA HABILITADA?
                GOTO MAINLOOP               ; NO
                CLR CONTX                           ;
                MOV #BUFFER,W4              ;
                ADD #0x0040,W4              ; W4=#BUFFER+512;
VUEL2:          MOV.B [W4++],W0             ;
                MOV.B WREG,U2TXREG          ; TRANSMITE BYTE
T2:             BTSS IFS1,#U2TXIF ; PRUEBA HASTA TRANSMITIR TODO
                GOTO T2                     ;
                BCLR IFS1,#U2TXIF           ; BORRA FLAG
                INC CONTX                           ; INCREMENTA CONTEO DE BYTES TRANSMITIDOS
                MOV #0x0040,W0              ; W0=512
                XOR CONTX,WREG              ; BYTES TRANSMITIDOS=512?
                BRA NZ,VUEL2                ; NO->CONTINUE TRANSMITIENDO
                GOTO MAINLOOP               ; SI->FINALICE

;==============================================================================
__OscillatorFail:
                        NOP                         ;
                        bra __OscillatorFail ;Loop forever when oscillator failure occurs
;------------------------------------------------------------------------------
;Address Error trap routine
__AddressError:
                        NOP                         ;
                        bra __AddressError ;Loop forever when address error occurs
;------------------------------------------------------------------------------
;Stack Error trap routine
__StackError:
                        NOP                         ;
                        bra __StackError ;Loop forever when stack error occurs
;------------------------------------------------------------------------------
;Math (Arithmetic) Error trap routine
```

```
__MathError:
                                NOP                             ;
                                bra __MathError ;Loop forever when math error occurs
;-----------------------------------------------------------------------------
;Math (Arithmetic) Error trap routine
__U2RXInterrupt:
                                PUSH W0                                 ;
                                BCLR IFS1,#U2RXIF               ; BORRA FLAG DE RECEPCION UART

                                MOV U2RXREG,W0                  ;
                                AND #0x00FF,W0                  ; BORRA LOS 8 MSBS DE U2RXREG
                                MOV W0,RXAUX                    ; ALMACENA U2RXREG EN RXAUX
                                MOV #0x00AA,W1                  ;
                                CPSEQ W0,W1                             ;
                                GOTO P1                                 ;
                                BSET FLAGS,#CARDEN              ;
                                GOTO P2                                 ;
                        P1:     MOV #0x0055,W1                  ;
                                CPSEQ W0,W1                             ;
                                GOTO P2
                                BCLR FLAGS,#CARDEN              ;

                        P2:     POP W0                                  ;
                                RETFIE
;----------------------------------------------------------------------------------------------
;T1 Interrupt routine
__T1Interrupt:
                        PUSH W0                                 ;
                        BCLR IFS0,#T1IF                 ;

                AD:     BTSS ADCON1,#DONE               ;
                        GOTO AD                                 ;

                        MOV #BUFFER,W1                  ;
                        MOV BYTESAV,W2                  ;
                        MOV ADCBUF1,WREG                ;
                        MOV WREG, VOLTAJE               ;
                        MOV W0,[W1+W2]                  ;
                        INC2 BYTESAV                    ;
                        MOV BYTESAV,W2                  ;
                        MOV ADCBUF2,WREG                ;
                        MOV WREG,CORRIENTE              ;
                        MOV W0,[W1+W2]                  ;
                        INC2 BYTESAV                    ;
                        MOV #0x0080,W0                  ;
                        CP BYTESAV                              ;
                        BRA NZ,F2                               ;
                        CLR BYTESAV                             ;
                        BSET FLAGS,#BFULL               ;
                F2:     BTSS FLAGS,#CARDEN
                        GOTO REST1

                        BTSC FLAGS,#IS
                        GOTO IN
                        MOV #0x0001,W0
                        MOV W0,LOAD1
                        CALL SUICH
                        BSET FLAGS,#IS
                        GOTO FIN

                        ; MPPT IP&O
                IN:     BTSC FLAGS,#H                   ;
                        GOTO HISTERESIS                 ;
                        BTSS FLAGS,#BR                  ;
                        GOTO N1                                 ;
```

```
                BTSS FLAGS,#BL          ;
                GOTO N2                         ;
                MOV VOLTAJE,W0          ;
                MUL.W CORRIENTE                 ;
                MOV W2,P2L                      ;
                MOV W3,P2H                      ;
                GOTO INICIO                     ;
        N2:     MOV VOLTAJE,W0          ;
                MUL.W CORRIENTE                 ;
                MOV W2,P1L                      ;
                MOV W3,P1H                      ;
                GOTO INICIO                     ;
        N1:     BTSS FLAGS,#BI          ;
                GOTO N3                         ;
                MOV VOLTAJE, W0         ;
                MUL.W CORRIENTE                 ;
                MOV W2,P2L                      ;
                MOV W3,P2H                      ;
                BSET FLAGS,#BR          ;
                GOTO INICIO                     ;
        N3:     MOV VOLTAJE,W0          ;
                MUL.W CORRIENTE                 ;
                MOV W2,P1L                      ;
                MOV W3,P1H                      ;
                MOV LOAD1,W0            ;
                ADD #0x0001,W0          ;
                AND #0x00FF,W0          ;
                MOV W0,LOAD2            ;
                CALL SUICH                      ;
                BSET FLAGS,#BI          ;
                GOTO FIN                        ;
INICIO:         MOV P1H,W0                      ;
                MOV P2H,W1                      ;
                CPSGT W1,W0                     ;
                GOTO PT1                        ;
                CPSEQ W0,W1                     ;
                GOTO PT2                        ;
                MOV P1L,W0                      ;
                MOV P2L,W1                      ;
                CPSGT W1,W0                     ;
                GOTO PT1                        ;
  PT2:          BTSS FLAGS,#B2          ;
                GOTO N4                         ;
                MOV P2H,W0                      ;
                MOV P2L,W1                      ;
                MOV W0,P1H                      ;
                MOV W1,P1L                      ;
                MOV LOAD1,W0            ;
                ADD #0x0001,W0          ;
                AND #0x00FF,W0          ;
                MOV W0,L                        ;
                CALL SUICH                      ;
                BSET FLAGS,#H           ;
                BCLR FLAGS,#BL          ;
                GOTO FIN                        ;
        N4:     MOV P2H,W0                      ;
                MOV P2L,W1                      ;
                MOV W0,P1H                      ;
                MOV W1,P1L                      ;
                MOV LOAD2,W0            ;
                ADD #0x0001,W0          ;
                AND #0x00FF,W0          ;
                MOV W0,LOAD2            ;
                CALL SUICH                      ;
                BSET FLAGS,#B1          ;
```

```
                    BSET FLAGS,#BL              ;
                    GOTO FIN                            ;
  PT1:              BTSS FLAGS,#B1              ;
                    GOTO N5                             ;
                    MOV LOAD2,W0                ;
                    ADD #0x0001,W0              ;
                    AND #0x00FF,W0              ;
                    MOV W0,L                            ;
                    CALL SUICH                          ;
                    BSET FLAGS,#H               ;
                    BSET FLAGS,#BL              ;
                    GOTO FIN                            ;
          N5:       MOV P1H,W0                          ;
                    MOV P1L,W1                          ;
                    MOV W0,P2H                          ;
                    MOV W1,P2L                          ;
                    MOV LOAD1,W0                ;
                    ADD #0x0001,W0              ;
                    AND #0x00FF,W0              ;
                    MOV W0,LOAD1                ;
                    CALL SUICH                          ;
                    BSET FLAGS,#B2              ;
                    BCLR FLAGS,#BL              ;
                    GOTO FIN                            ;
;-------------------------------------------------------------------------------------------
HISTERESIS:                 MOV VOLTAJE,W0              ;
                            MUL.W CORRIENTE                     ;
                            MOV #0x039E,W1              ;
                            CPSGT W0,W1                         ;
                            GOTO REST                           ;
                            MOV W2,PL                           ;
                            MOV W3,PH                           ;
                            MOV P1H,W4                          ;
                            BCLR SR,#N                          ;
                            SUB W4,W3,W1                ;
                            BTSS SR,#N                          ;
                            GOTO N6                             ;
                            MOV PH,W0                           ;
                            MOV PL,W1                           ;
                            MOV W0,P1H                          ;
                            MOV W1,P1L                          ;
                            MOV L,W0                            ;
                            MOV W0,LOAD1                ;
                            ADD #0x0001,W0              ;
                            AND #0x00FF,W0              ;
                            MOV W0,L                            ;
                            CALL SUICH                          ;
                            GOTO FIN                            ;
                    N6:     MOV #0x0004,W0              ;
                            CPSGT W1,W0                         ;
                            GOTO N7                             ;
                            MOV #0x0007,W0              ;
                            CPSGT W1,W0                         ;
                            GOTO N8                             ;
                            GOTO REST                           ;
                    N7:     MOV L,W0                            ;
                            AND #0x00FF,W0              ;
                            CALL SUICH                          ;
                            GOTO FIN                            ;
                    N8:     MOV L,W0                            ;
                            SUB #0x0008,W0              ;
                            AND #0x00FF,W0              ;
                            MOV W0,LOAD1
                            CALL SUICH                          ;
                            CLR FLAGS                           ;
```

```
                    BSET FLAGS,#CARDEN      ;
                    GOTO FIN                        ;
REST:               CLR FLAGS                       ;
                    BSET FLAGS,#CARDEN      ;
                    ;MOV #0X0001,W0         ;
                    ;MOV W0,LOAD1           ;
                    ;CALL SUICH                     ;
                    GOTO FIN                        ;
REST1:              CLR FLAGS                       ;
                    CLR W0                          ;
                    CALL SUICH                      ;


FIN:        POP W0                          ;
            RETFIE                          ;
;==============================================================================
SUICH:          MOV W0, ACBYTE
                MOV ACBYTE,WREG                 ; PALABRA 2-BYTES
                MOV WREG,REGIS1                 ;
                RLNC REGIS1                     ;
                RLNC REGIS1                     ;
                RLNC REGIS1,WREG        ;
                MOV WREG,LATB           ;
                MOV ACBYTE,WREG                 ;
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                MOV W0,REGIS1           ;
                MOV #0x000C,W0          ;
                AND REGIS1                      ;
                MOV ACBYTE,WREG
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                RRNC W0,W1                      ;
                RRNC W1,W0                      ;
                MOV W0,REGIS2           ;
                MOV #0x0300,W0          ;
                AND REGIS2,WREG         ;
                IOR REGIS1,WREG         ;
                MOV WREG,LATD           ;
                BTSS ACBYTE,#15         ;
                GOTO SALTO1                     ;
                BSET LATA,#11           ;
                GOTO SALTO2                     ;
SALTO1:         BCLR LATA,#11           ;
SALTO2:         RETURN                          ;


        .end ;End of code in this file
```

APPENDIX B SOFTWARE

The following program and subprograms create the graphical user interface which the user can to control the start of tracking for the Selective Hopping and Improved Perturbation and Observation methods. Additionally, they perform the communication protocol between the PC and the embedded system.

```
;==========================================================================
;==========================================================================
function varargout = VisualMPPT(varargin)
% VISUALMPPT M-file for VisualMPPT.fig
%      VISUALMPPT, by itself, creates a new VISUALMPPT or raises the existing
%      singleton*.
%
%      H = VISUALMPPT returns the handle to a new VISUALMPPT or the handle to
%      the existing singleton*.
%
%      VISUALMPPT('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in VISUALMPPT.M with the given input arguments.
%
%      VISUALMPPT('Property','Value',...) creates a new VISUALMPPT or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before VisualMPPT_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to VisualMPPT_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help VisualMPPT

% Last Modified by GUIDE v2.5 25-Oct-2006 18:40:12

% Begin initialization code - DO NOT EDIT
```

```
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @VisualMPPT_OpeningFcn, ...
                   'gui_OutputFcn',  @VisualMPPT_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before VisualMPPT is made visible.
function VisualMPPT_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to VisualMPPT (see VARARGIN)
    global buff conb port i E f T h x1 x2 Nh

    T=0.3;                  % PERIODO DE MUESTREO (s)
    fs=4*(1/T);
    fc=1.2*(1/T);           %RESPUESTA AL IMPULSO FILTRO PASABAJOS
    fn=2*pi*(fc/fs);
    n1=0:12;
    n=n1-6;
    hd=(fn/pi)*sinc((fn/pi)*n);
    Nh=length(hd);
    h=hd.*hamming(Nh)';
    x1=zeros(1,Nh);
    x2=zeros(1,Nh);
    c1=zeros(1,16);
    v1=zeros(1,16);

    set(handles.figure1,'CloseRequestFcn','closecallback')
```

```
conb=0;
i=0;
E=0;
f=0;

xlabel(handles.axes1,'Time (s)');
ylabel(handles.axes1,'Power (W)');
title(handles.axes1,'Power');
grid(handles.axes1,'on');
hold(handles.axes1,'on');

xlabel(handles.axes2,'Time (s)');
ylabel(handles.axes2,'Voltage (V)');
title(handles.axes2,'Voltage');
grid(handles.axes2,'on');
hold(handles.axes2,'on');

xlabel(handles.axes3,'Time (s)');
ylabel(handles.axes3,'Current (A)');
title(handles.axes3,'Current');
grid(handles.axes3,'on');
hold(handles.axes3,'on');

 port = serial('COM1');
% configuracion del puerto serial
port.BaudRate=115200;
port.DataBits = 8;
port.DataTerminalReady = 'off';
port.FlowControl = 'none';
port.parity='none';
port.ReadAsyncMode = 'continuous';
port.StopBits = 1;

port.ByteOrder = 'littleEndian';
port.OutputBufferSize =3000000;
port.Timeout =320;
port.BytesAvailableFcnCount = 64;
port.BytesAvailableFcnMode = 'byte';
port.BytesAvailableFcn =  {@bufferfullcallback,handles};
port.InputBufferSize = 64;
%get(port) Habilita puerto
fopen(port);
```

```
    %confiruracion funcion timer



% Choose default command line output for VisualMPPT
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes VisualMPPT wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = VisualMPPT_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global i port

if i==0;
fwrite(port,170,'uint8','sync'); %Transmite palabra de Inicio de Flujo de Datos 0xAA
set(handles.pushbutton1,'String','Stop');
i=1;
else
fwrite(port,85,'uint8','sync'); %Transmite palabra de Inicio de Flujo de Datos 0x55
set(handles.pushbutton1,'String','Start');
i=0;
end

 % --- callback del buffer.
```

```
function bufferfullcallback (obj,event,handles)

    global port conb E f T h x1 x2 Nh

out = fread(port,port.BytesAvailable,'uint8');

f=f+1;
if f==142 %3 min de despliegue graficas
conb=0;
f=0;
cla(handles.axes1);
cla(handles.axes2);
cla(handles.axes3);
end

%T=0.160;
ViL=1:4:64;
ViH=2:4:64;
IiL=3:4:64;
IiH=4:4:64;

for i=1:16
  x1(16+i)=bitor( out(ViL(i)),bitshift(out(ViH(i)),8) )*4.8/4096;
  x2(16+i)=bitor( out(IiL(i)),bitshift(out(IiH(i)),8) )*44.2/4096 ;
end


for i=1:16
  a=sort(x1(i:15+i))
  b=sort(x2(i:15+i))
   corriente(i)=a(8);
   voltaje(i)=b(8);    %filtro de medias
end

for i=1:16
    x1(i)=x1(16+i);
    x2(i)=x2(16+i);
end

corriente;
voltaje ;

potencia=voltaje.*corriente;
```

```
plot(handles.axes1,( 0:T:(T*15) )+16*T*conb,potencia,'b:.');
plot(handles.axes2,(0:T:(T*15))+16*T*conb,voltaje,'m:.');
plot(handles.axes3,(0:T:(T*15))+16*T*conb,corriente,'r:.');
E=E+(T*sum(potencia))/3600;
if E>99999.99999
  E=E-99999.99999;
end
set(handles.text1,'String',num2str(E));
conb=conb+1;


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global conb

conb=0;

cla(handles.axes1);
cla(handles.axes2);
cla(handles.axes3);
```