SoC-FPGA Implementation of a Temperature-Dependent Parameters Estimator for Photovoltaic Generators

Ronny Zárate-Ferreto*, Kaleb Alfaro-Badilla*, Carlos Meza-Benavides*,
Carlos Salazar-García†, and Alfonso Chacón-Rodríguez *
*Ingeniería Electrónica, † Ingeniería Mecatrónica, Cartago, Instituto Tecnológico de Costa Rica
Corresponding Author: alchacon@tec.ac.cr

***Abstract*—This paper presents the hardware implementation of a linear estimation algorithm for the online monitoring of temperature-dependent parameters from photovoltaic (PV) systems. Such parameters are related to the optimal energy efficiency in PV panels, and the validity of the estimation algorithm has been verified in prior publications. The algorithm requires the sampling of high-frequency harmonics in the DC-DC converted voltage and current signals (in the order of several hundred kHz), which make their acquisition and processing unwieldy for typical small embedded systems, mainly if aiming at low latency, efficiency control loops for several panels. As such, the presented system provides with fast sampling (1Mbps per channel), fast processing, expandable parametric estimation, and online monitoring, using an Avnet ZedBoard Zynq SoC platform. Monitoring data from the system may be accessed via the Internet, and the low resources count of the implemented system opens the possibility for the incorporation of control algorithms running on hardware as well.**

***Index Terms*—System-on-chip, FPGA, heterogeneous computating, high-level synthesis, photo-voltaic power systems.**

I. Introduction

Grid-connected PV structures provide cost-effective, renewable energy, but such systems are susceptible to non-linear disturbances from environmental conditions such as cloudy weather. The most natural solution to stabilize a grid-PV system is adding a battery to store the produced energy. Unfortunately, batteries degrade the system's reliability while increasing maintenance costs.

Figure 1 shows the general system architecture of a smart PV power array, connected to a power utility. An online monitoring system may be connected to the current and voltage (I_{pv}, V_{pv}) outputs of each panel, providing thus for access to these panel's energy variables through a web app. But, although embedded solutions based on this scheme may exist, network latency and the embedded acquisition system and processing speed do not allow for real-time I_{pv} and V_{pv} monitoring, let alone deriving intrinsic the PV modules' characteristics from such variables, or establishing a high-speed closed-loop control application (with a target latency bounded by a few microseconds), even if placing the control algorithm inside the embedded system itself. Solutions for a control-device that keeps the overall power output of the grid-PV at a steady-state have been proposed, yet the parameters required to measure the panel variables are not trivial to obtain (see [1], [2]). Trying to avoid the computational cost of such analytic algorithms, some authors propose the use heuristic algorithms for achieving real-time execution, with mixed results (several of these solutions are summarized in [2]). As an alternative, Meza and Ortega have proposed a straightforward analytical method that estimates such parameters by a continuous measurement of the current and voltage of each PV panel [3], [4]. The method, at a first glance, seems feasible to implement in a typical processor-based embedded system, with convergence speed depending on the input signals frequency (a consequence of the DC-DC conversion), entailing the high-speed acquisition and processing times. But even if disregarding hardware costs, a high-speed general processor with the necessary data acquisition devices would be at most capable of handling a few variables before surpassing the required latency bounds. FPGAs, on the other hand, due to their inherent parallelism, can run high-speed computation structures concurrently, but their programming is somewhat complicated, requiring besides extra modules for handling user interfaces and data communications. A heterogeneous processing architecture, such as Xilinx SoC-FPGA Zynq 7000 family, mixes the best of both worlds, delivering parallel processing with ease of development (through the use of tools such as Vivado's high-level synthesis suite), and a standard programming software development kit with the advantages of standard interfaces for bare-metal or operating system-based solutions. Accordingly, this work proposes a HLS-based, heterogeneous hardware implementation for such an estimator for PV panel parameters, based on the method described in [3], [4], using an Avnet's Zynq-7020 SoC ZedBoard with 512MB DDR3 RAM.

This paper is organized as follows: Section II provides a quick overview of the theoretical foundations of the designed PV parameter estimator. Section III provides a high-level overview of the proposed design, and describes the HLS methodology applied for implementing the needed IP blocks for each required function. Section IV details the hardware-software heterogeneous co-design strategy followed. Section

978-1-7281-3427-7/20/$31.00 ©2020 IEEE

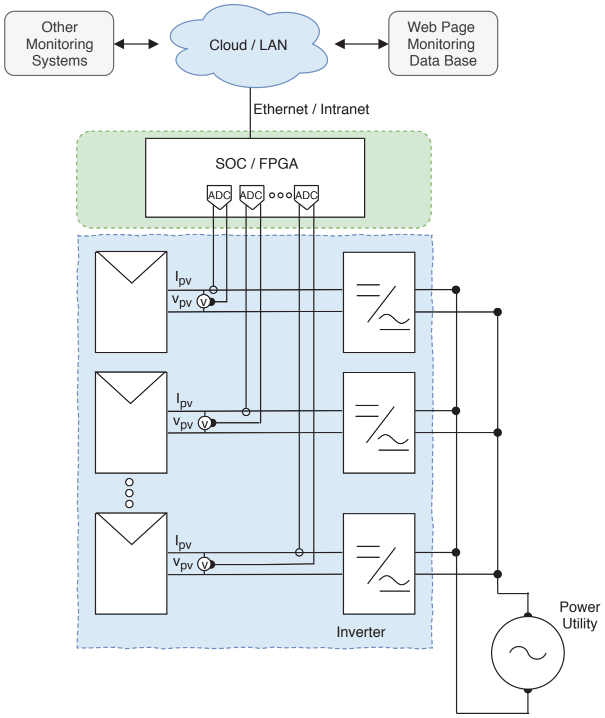

Figure 1. High level architecture of a smart photo-voltaic micro-grid power system, remotely monitored.

V shows the results of the proposed system's performance. Finally, conclusions and recommendations for future improvements are given in Section VI.

II. Theoretical Background

As explained in [3], [4], the output current provided by a PV module may be described by a one-diode electrical model, i.e.,

$$I_{pv} = I_g(S) - I_s(T)\exp[\alpha(T)V_{pv}] \tag{1}$$

where I_g is the PV panel short-circuit current (dependent on S, the solar incident radiation), and $\alpha = \theta_1$ and $I_S = \theta_2$ are both temperature-related variables that may be estimated according to [3], [4] by applying a gradient-based estimator to the linearized model of (1), such that

$$\dot{\hat{\theta}}(t) = \Gamma\Phi(t)[y(t) - \Phi(t)^T\hat{\theta}(t)] \tag{2}$$

where $\Phi(t) = [V_{pv}(t), 1]^T$ is the vector holding the voltage signal read from the PV panel; $\hat{\theta} = [\hat{\theta}_1, \hat{\theta}_2]^T$ is the estimated parameters vector; Γ is the 2×2 constant coefficient matrix; and $y(t)$ is the linearizing of the difference between I_g and the PV panel output current, namely:

$$y(t) = \ln\left[I_g - I_{pv}(t)\right] \tag{3}$$

A discreet approximation for Eq. (2) is given by the first-order Euler's method, such that

$$\dot{\hat{\theta}}_{n+1} = \Gamma\Phi_n(y_n - \Phi_n^T\hat{\theta})T_s + \dot{\hat{\theta}}_n \tag{4}$$

with T_s the sampling time.

III. General Hardware Design Strategy

Due to its availability and prototyping capabilities, an Avnet Zedboard, based on a Xilinx Zynq-7020 All-Programmable SoC, was chosen (see [5] for technical details). A general overview of the proposed system is given in Fig. 2.

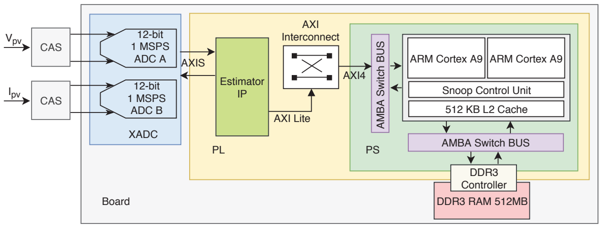

Figure 2. Overall system architecture. The ADCs send samples via AXI-Stream to the implemented IP, which solves the estimator equation for every sample. The resulting accumulative parameters are communicated to the PS via AXI4Lite.

The Zynq-7020 has two integrated ADCs that are used to sample the I_{pv} and V_{pv} signals, previously conditioned through an analog circuit to the adequate ADC's input levels (1V), from the PV generator. A high-level view of the dataflow scheme proposed for the EU is shown in Fig. 3. The estimator-unit (EU), located in the SoC's FPGA programmable region (also known as PL in Xilinx's documentation), receives measurements via a FIFO AXI-Stream interface. The proposed EU is made of three operational blocks: `ADC-to-Real`, `Linearization-Unit` and `Parameters-Estimator`. The ADCs stream feeds data to the EU at a constant rate, with the EU updating estimations at the defined sampling rate. The first block captures the vector pair composed of I_{pv} and V_{pv}, re-scaling them to their real values. The second block solves Eq.(3). The third block performs the estimation algorithm itself. Estimated parameters are made available for reading to the ARM via a memory-mapped bus interface (AXI-Lite). Using network services, is it be possible to provide remote monitoring of such parameters, using a platform as the one reported by this paper's authors in [6].

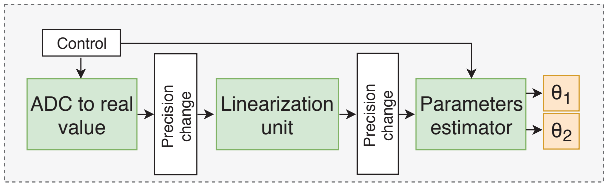

Figure 3. Proposed dataflow scheme for the Estimator Unit (EU) hardware IP.

These operational units are coded in C++, using a guided high-level synthesis (HLS) methodology, as recommended in [7], and already tested by this papers' authors in previous works (see [8], [9]). The C++ HLS coding style focuses on various principles: each unit-block is coded as a task, with each task synchronized via FIFO interfaces. C++ template data types are used to make tasks type-agnostic, allowing for fast design-space exploration in terms of the adequate numeric representation and precision, and speeding up as well the cross-evaluation of alternative algorithmic implementations. Finally, each task is optimized in performance and resource utilization individually, where having configurable numeric precision allows for optimal performance and resource utilization (as later discussed in Section V). Unit-blocks in Fig. 3 are therefore directly translated into functions receiving as arguments the FIFO structures. The following subsections give the implementation details of each block.

A. *Parameter-estimator design*

The estimation of the intended PV parameters requires solving the differential equation given in Eq. (2). The EU receives y_n and V_{pv}, and solves the matrix operations in Eq. (4). A Q11.21 fixed-point, two's complement representation was used. Accuracy was arrived at after a thorough space design exploration of the estimator's high-level arbitrary precision C++ model, aiming at a bounded 1% error against the floating-point Python estimator's model (as given by [3], [4]).

B. *Linearization-Unit implementation*

The linearizing in Eq. (3) requires applying a natural logarithm. Since the Xilinx LogiCore IP library does not provide a fixed-point logarithm function, a fixed-point logarithm approximation was implemented, based on the *fast binary logarithm algorithm* presented in [10], using a Q4.13 two's complement fixed-point representation. The algorithm's precision depends on the number of iterations; here, iterations were fixed to those achieving a bounded error under 1% for the measured dynamic range of I_{pv}. Figure 4 shows the approximation's results and those of Python's standard floating-point natural logarithm operations, with error computed as $err = (o - \hat{o})/o$, where o is the Python's operation value and $\hat{o}$ the approximation used.

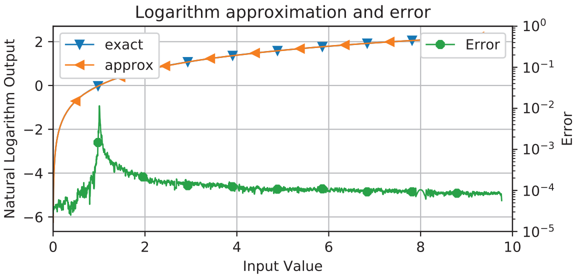

Figure 4. The natural logarithm approximation used (see [10]), using a Q5.13 two-complement fixed-point representation. Results are compared against Python's standard floating-point natural logarithm operations. The error is bounded under 1% for the measured dynamic range of I_{pv}.

IV. System Implementation and Experimental Validation

The Zedboard has only two built-in ADC modules, but more acquisition channels may be added via the Zedboard Pmod extensions, or multiplexing the ADCs, using lower sampling speeds if feasible. Besides, being this a flexible design, porting it to other Zynq boards with larger resources, is straightforward. Each ADC is 12-bit, with a maximum 1 Msps sampling speed. Each sample is stored in a FIFO interface which is read by the EU. The EU runs at 100MHz, taking 41 cycles to process one I_{pv}-V_{pv} pair (i.e., 410ns). This means that the system can be real-time compliant, since $T_{compute} < T_s$. The configuration setup of the EU is performed by the ARM, through the AXI-Lite bus interface. Resources usage for the Zedboard based version here presented is of 41 DSPs (19%), with 2426 Flip-Flops (6%) FF and 5993 LUTs (9%) required. Table I compares resource utilization taken from the Vivado synthesis report for three different targets. Note that Zynq Z-7007S can manage the processing of one PV panel, whereas midrange FPGAs can accommodate from 5 to 20 EUs.

Table I
PL resource utilization comparison for three different Zynq®-7000 SoC Family models, for a single estimator unit. Last row shows how many potential EUs in parallel may be fitted on each model.

Resource		Utilization (%)		
Name	Total	Z-7007S	Z-7020	Z-7035
DSP	41	62	19	5
FF	2426	24	6	2
LUT	5993	35	9	3
Potential EUs		1	5	20

In order to validate the system, a simulation test scenario with known test signals was set up. The overall validation proposal is shown in Fig. 5. A signal pattern (generated with data from a Kyocera KC65T commercial PV module, according to the model used in [4]) was synthesized using Python, and reproduced through a PC's sound card (using the L-R channels for the current and voltage signals respectively at 192ksps) which is sufficient to reproduce the PV characteristics. As explained before, signals are then filtered and scaled in an analog conditioning circuit, before being fed to the ADCs.

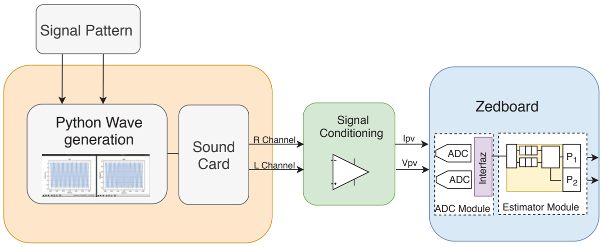

Figure 5. Experimental setup used for validating the system. A signal synthesizer was used for the creation of the input I_{pv} and V_{pv} signals. See Fig. 6.

V. Results

The EU is configured with the parameters shown in Table II. The $\hat{\theta}$ values are sampled from the EU by the ARM, and after 15 seconds of continuous measurements, data is collected via the UART port of the Zedboard. This dataset is plotted and compared against Python simulation results in Fig. 7; error is bounded under 2%.

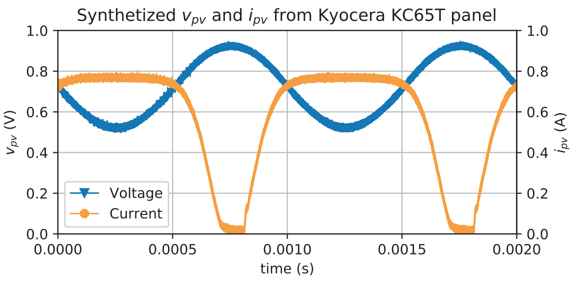

Figure 6. Current (I_{pv}) and voltage (V_{pv}) test inputs used for the system validation stage. Signal pattern characteristics were generated according to parameters from a commercial Kyocera KC65T solar panel [11], using the model given in [3].

Table II
PARAMETERS USED FOR THE ESTIMATOR SETUP, AS PROPOSED IN [3], [4].

Parameter	γ_{11}	γ_{12}	γ_{21}	γ_{22}	$\hat{\theta}_1(t=0)$	$\hat{\theta}_2(t=0)$
Value	0.1	0	0	100	0,55	-13

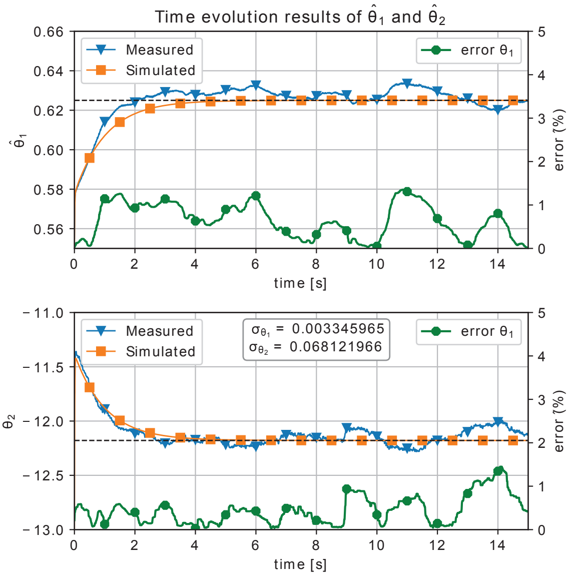

Figure 7. Time convergence of $\hat{\theta}_1$ and $\hat{\theta}_2$. Measurements were taken from the system shown in 5, and the simulation results from the Python-based model. Dashed lines serve as the theoretical expected values of $\hat{\theta}_1$ and $\hat{\theta}_2$. The error is plotted at the bottom subfigure. The error standard deviation is bounded under 0.07.

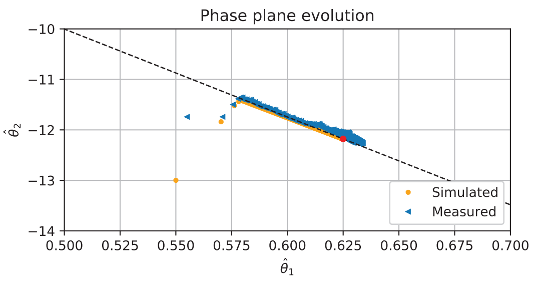

Figure 8. Convergence of $\hat{\theta}_1$ and $\hat{\theta}_2$ in the phase plane. The dashed line embody the nullcline. The red circle points locate where $\{\hat{\theta}_X - \theta_X\}$=0.

VI. CONCLUSIONS

A photovoltaic generator parameter estimator has been proposed and validated with synthesized signals derived from a commercial PV panel model, with an error bounded under 2%. The given SoC-FPGA heterogeneous implementation met the intended latency computational time (well under half the specified $1\mu s$ sampling rate), which makes it suitable for its future integration into real-time PV closed-loop control systems. The resulting estimator unit is also compact and portable, hence allowing for the management of multiple PV panels by a single SoC-FPGA board. Its standard AXI interfacing to a general ARM based embedded platform facilitates the estimator's integration into an Internet-based monitoring network.

REFERENCES

[1] C. Meza, D. Jeltsema, J. Scherpen, and D. Biel, "Passive p-control of a grid-connected photovoltaic inverter," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 5575 – 5580, 2008, 17th IFAC World Congress. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667016398330

[2] A. R. Jordehi, "Parameter estimation of solar photovoltaic (PV) cells: A review," *Renewable and Sustainable Energy Reviews*, vol. 61, pp. 354–371, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.rser.2016.03.049

[3] C. Meza and R. Ortega, "Control and estimation scheme for PV central inverters," in *2013 XXIV International Conference on Information, Communication and Automation Technologies (ICAT)*, Oct 2013, pp. 1–6.

[4] C. Meza and R. Ortega, "On-line estimation of the temperature dependent parameters of photovoltaic generators," *IFAC Proceedings Volumes*, vol. 46, no. 11, pp. 653 – 658, 2013, 11th IFAC Workshop on Adaptation and Learning in Control and Signal Processing. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S147466701633018X

[5] Xilinx Inc. Zynq-7000 All Programmable SoC First Generation Architecture. [Online]. Available: https://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf

[6] L. G. León-Vega, K. Alfaro-Badilla, A. Chacón-Rodríguez, and C. Salazar-García, "Optimizing big data network transfers in FPGA SoC clusters: TECbrain case study," *High Performance Computing*, 2019, 6th Latin American Conference, CARLA 2019.

[7] J. de Fine Licht, S. Meierhans, and T. Hoefler, "Transformations of high-level synthesis codes for high-performance computing," *CoRR*, vol. abs/1805.08288, 2018. [Online]. Available: http://arxiv.org/abs/1805.08288

[8] K. Alfaro-Badilla, A. Chacón-Rodríguez, G. Smaragdos, C. Strydis, A. Arroyo-Romero, J. Espinoza-González, and C. Salazar-García, "Prototyping a biologically plausible neuron model on a heterogeneous CPU-FPGA board," in *2019 IEEE 10th Latin American Symposium on Circuits Systems (LASCAS)*, Feb 2019, pp. 5–8.

[9] K. Alfaro-Badilla, A. Arroyo-Romero, C. Salazar-García, L. G. León-Vega, J. Espinoza-González, F. Hernández-Castro, A. Chacón-Rodríguez, G. Smaragdos, and C. Strydis, "Improving the simulation of biologically accurate neural networks using data flow HLS transformations on heterogeneous SoC-FPGA platforms," *High Performance Computing*, 2019, 6th Latin American Conference, CARLA 2019.

[10] C. S. Turner, "A fast binary logarithm algorithm [dsp tips tricks]," *IEEE Signal Processing Magazine*, vol. 27, no. 5, pp. 124–140, Sep. 2010.

[11] Kyocera Corporation. KC65T High Efficiency Multicrystal Photovoltaic Module. [Online]. Available: https://www.kyocerasolar.com/dealers/product-center/archives/spec-sheets/KC65T.pdf