HarvWSNet: A co-simulation framework for energy harvesting wireless sensor networks

Amine Didioui, Carolynn Bernier, Dominique Morche, Olivier Sentieys

HarvWSNet: A Co-Simulation Framework for Energy Harvesting Wireless Sensor Networks

Amine Didioui[1,2], Carolynn Bernier[1], Dominique Morche[1], and Olivier Sentieys[2]
[1]CEA/LETI - Minatec, Grenoble, France
[2]IRISA/INRIA - University of Rennes 1, Lannion, France

***Abstract*—Recent advances in energy harvesting (EH) technologies now allow wireless sensor networks (WSNs) to extend their lifetime by scavenging the energy available in their environment. While simulation is the most widely used method to design and evaluate network protocols for WSNs, existing network simulators are not adapted to the simulation of EH-WSNs and most of them provide only a simple linear battery model. Therefore, there is a need for a framework suited to EH-WSN simulation and to lifetime prediction. We propose a co-simulation framework, HarvWSNet, based on WSNet and Matlab, that provides adequate tools for the simulation of the network protocols and the lifetime of EH-WSN. Indeed, the framework allows for the simulation of multi-node network scenarios while including a detailed description of each node's energy harvesting, management subsystem and its time-varying environmental parameters. A simulation case study based on a temperature monitoring application demonstrates HarvWSNet's ability to predict network lifetime while minimally penalizing simulation time.**

***Index Terms*—Energy harvesting, wireless sensor networks, co-simulation, WSNet, Matlab.**

I. Introduction

Wireless sensor networks are typically composed of a large number of small and low-cost sensor nodes that are able to communicate over a radio channel but have limited energy supply and processing power. Application fields include environmental monitoring, structural health monitoring (e.g. automotive, healthcare, smart buildings, etc.). Currently, most WSNs rely on a limited source of energy such as batteries which must eventually be replaced or recharged. For example, using a high capacity AA battery of 3000 mAh, a Crossbow MICAz mote operating at a duty cycle of 1% lasts a maximum of 17.35 months [1]. In practice, replacing batteries is usually expensive, due to large network size, and often difficult or impractical in inaccessible or hostile environments.

Recent advances in energy harvesting technologies allow battery-powered nodes to extend their lifetime by scavenging energy from the environment. The most common sources of energy harvesting are solar, thermal and mechanical vibration energy [2]. However, the energy delivered by these sources is non-constant and depends on a variety of factors. To evaluate the feasibility of an applicative scenario, both the energy harvesting subsystem and the networking protocols must be validated simultaneously. A first approach consists in building a prototype network using existing hardware platforms. This provides accurate data about the system but can be very expensive, time-consuming and limited to small scale networks and existing hardware. To avoid these difficulties, simulations are commonly used in WSN research.

To the best of our knowledge, most WSN simulators deal only with discrete-time models which are not well adapted to the simulation of continuous-time systems such as energy harvesting. Moreover, most network simulators offer only a linear battery model [3] where the energy is decremented at each power consuming event. It is difficult, using such a simple model, to account for the complex behavior of an energy harvesting unit also including the storage devices. Modeling such a unit requires taking into account the time varying environmental parameters, the harvester and converter efficiencies, the power management algorithms and the battery charge and discharge behavior. In addition, the network and energy harvesting events may be completely uncorrelated. For these reasons, this work proposes a co-simulation framework, HarvWSNet, which combines the strengths of two different tools, namely MATLAB and WSNet, to produce more realistic simulation results for the evaluation of energy harvesting wireless sensor networks (EH-WSN) performance and lifetime.

The MATLAB environment is used for modeling the energy harvesting system and for data visualization while WSNet [4], a precise wireless sensor network simulator, is used to implement the communication protocol layers and simulate the network behavior. The simple battery model of WSNet is replaced by the energy harvesting system implemented in MATLAB. Both MATLAB and WSNet are run interactively and exchange data using sockets.

The energy harvesting system of HarvWSNet is completely modular and can adapt to any type of harvested energy from simple to more complex models. In this paper, we describe a solar energy harvesting model based on the datasheets of COTS devices. Using the new co-simulation platform described in this paper, it is possible to simulate the network behavior for several weeks, months or years using the environmental data of a specific location in the world, hence easing the design of a wide-range of novel EH-WSN deployment scenarios.

This paper is organized as follows. In the next section, we give an overview of state-of-the-art co-simulation frameworks. In Section III, we present the co-simulation framework HarvWSNet. Section IV describes our solar energy harvesting model. Finally, we evaluate the performance and efficiency of HarvWSNet using a case study in Section V.

978-1-4673-5288-8/13/$31.00 ©2013 IEEE

II. Related Work: Wireless Sensor Network Co-simulation Frameworks

In recent years, several co-simulation frameworks have been proposed in order to simulate the network and physical systems together. Network simulators such as WSNet, OMNet++, and NS-2 are used to simulate the network protocol stack whereas physical systems simulators such as Matlab/Simulink, Modelica and SystemC are dedicated to the simulation of systems combining continuous and discrete dynamics.

COSMO [5] is a co-simulation framework based on Matlab and OMNet++ which was developed to study the impact of 3-D indoor channels on wireless networks. Channel models developed in Matlab are compiled directly into shared libraries and integrated into OMNet++. However, COSMO is not adapted to the simulation of EH-WSN requiring interactive simulation and clock synchronization between the WSN simulator and the energy harvesting system. Piccsim [6], combining Simulink and NS-2, is a co-simulation framework for wireless networked control systems (WNCS). In this framework, Simulink and NS-2 are run on different computers and coordinated by TCP/UDP packets using the Piccsim Toolchain. Finally, the OPNET-SIMULINK co-simulation platform [7] combines OPNET and Matlab to study the effect of node movement on WNCS. Similarly to Piccsim, this platform requires two computers, the first running OPNET and the second running Matlab. Both simulators are executed in parallel and are synchronized using UDP sockets. In both of these co-simulation platforms, much development effort is needed to implement the synchronization mechanism between the two computers. The co-simulation framework proposed in this work, HarvWSNet, employs a simple synchronization process between WSNet and Matlab and requires only one computer. In addition, unlike all of the co-simulation frameworks presented above, our platform is based on WSNet because only this network simulator offers sufficiently detailed models of the radio channel and physical layer [8].

III. Co-simulation with WSNet and MATLAB

A. Co-simulation framework

As stated previously, the goal of HarvWSNet is to evaluate the power charge/discharge profile of energy harvesting nodes in realistic scenarios. To this end, the simple energy model of WSNet is replaced by an energy harvesting system developed in Matlab. This system is able to estimate accurately the energy harvested and the available energy in the storage element as a function of the environmental data and the current consumed by the node. For speed, the Matlab model is compiled into MEX function and this function is called directly by WSNet. Since WSNet does not provide any facility to directly connect to Matlab, we have built an interface using TCP sockets on both sides which allows WSNet and Matlab to exchange data periodically. The global architecture of the proposed framework is represented in Fig. 1.

The simulation is controlled by WSNet which defines the master clock and is responsible for clock synchronization. Even if, in this work, Matlab is used to model only the energy harvesting subsystem, the link between WSNet and Matlab

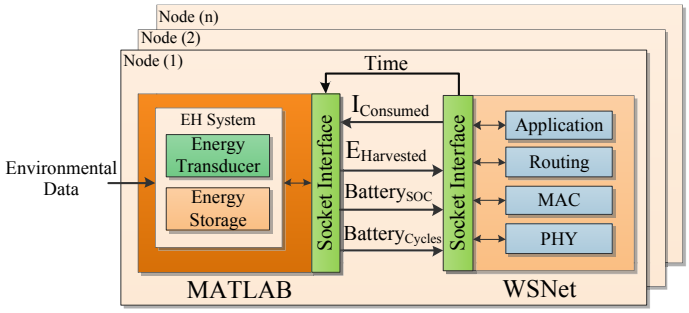

Fig. 1. Structure of HarvWSNet.

can be exploited for other tasks, e.g. to simulate complex channel models or elaborate RF transceiver models or for data visualization and hence open up a wide range of additional capabilities for WSNet.

B. Co-simulation process

An energy harvester is a dynamic system in which the battery state-of charge (SOC) and energy harvested change in a continuous manner with respect to time. On the other hand, WSNet is a discrete-event driven simulator that handles events that are unevenly distributed in time due to the stochastic nature of packet generation. We therefore define two types of events at which WSNet invokes Matlab. The first, called *synchronization events*, is scheduled at fixed intervals and dedicated to maintaining the synchronization of the simulation clocks between WSNet and Matlab. The second type of events, called *communication events*, arise when a node is in one of the following states: transmitting, receiving and listening.

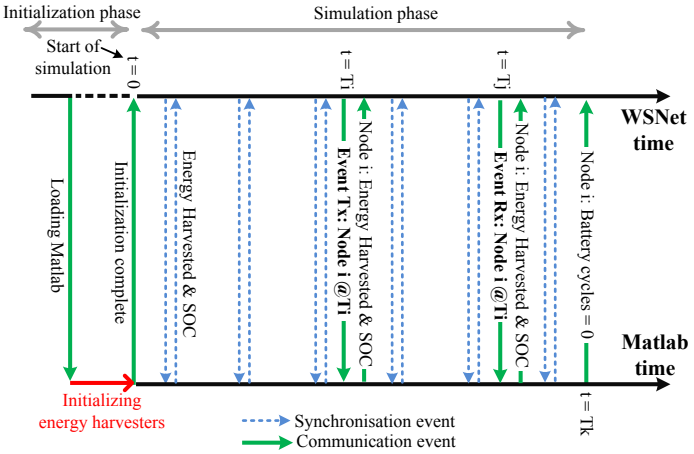

Fig. 2. Interactive co-simulation of WSNet and MATLAB.

Fig. 2 provides an overview of a typical exchange between WSNet and Matlab. During the initialization phase which takes place at the initial time but before the actual simulation starts, WSNet creates the nodes specified in the simulation configuration file and requests a connection to the Matlab engine via sockets. Once the connection is setup, Matlab initializes the energy harvesting system that corresponds to each simulated node and associates the predefined environmental data in accordance with the location of each node. After initialization,

WSNet begins the simulation at time zero and passes the clock to Matlab each T seconds allowing the energy harvester of each node to calculate the scavenged power, the battery SOC and the number of remaining charge/discharge cycles. The period T is controlled by WSNet and is calculated as a function of the communication duty-cycle and the environmental data type. When a node transmits or receives a packet, WSNet sends a notification to Matlab and waits until the Matlab calculation is done before continuing. The notification includes the node number, the kind of communication (reception or transmission), the packet duration, and the current to draw from the battery. Otherwise, if the node is not receiving nor transmitting data, the energy harvesting system considers that the node is in idle mode and draws the current corresponding to this state. When the battery associated to a node reaches its maximum number of charge/discharge cycles, Matlab sends a notification to WSNet indicating that the node is not operational and should be removed from the network.

IV. Energy Harvesting System Design

The aim of this section is to develop a model that not only is as realistic as possible but also that closely resembles existing energy harvesting platforms. The flexibility of the HarvWSNet framework eases the development of highly complex models that can be refined to take into account advanced power management strategies. Since the applicative scenario that will be described below is temperature monitoring application in a greenhouse, the energy scavenger chosen is a solar harvester (Fig. 3) consisting of a photovoltaic (PV) panel, a battery and a power manager. Thus, the environmental data used are solar irradiance and temperature. This information is loaded at the initialization phase (Fig. 2).

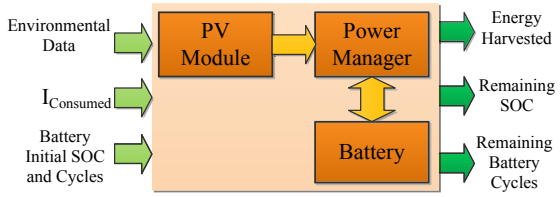

Fig. 3. Global architecture of the energy harvesting system.

When MATLAB receives a synchronization or a communication event, the solar harvester system calculates the solar energy scavenged, the state-of-charge (SOC) and remaining cycles of battery as function of the environment data, the current drawn by the node, the communication duration and type, and both the initial SOC and battery cycles. The models for the PV panel, battery and power manager will be described next.

A. Solar harvester modeling

Using the irradiance and temperature data, the solar harvester can estimate accurately the current and voltage that is delivered to the sensor node or to the battery. Fig. 4 represents the circuit diagram of the solar harvester which consists of a photovoltaic panel and a maximum power point (MPP) block.

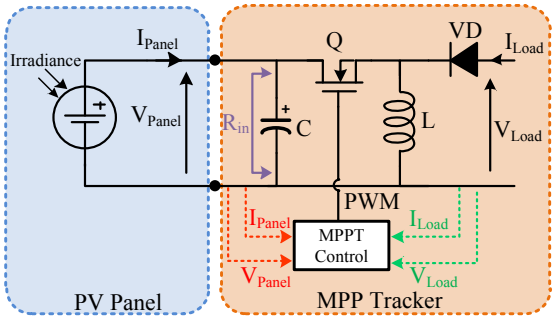

Fig. 4. Block diagram of solar energy harvesting system.

The PV panel *MULTICOMP MC-SP0.8-NF-GCS* provides a nominal output power of 800 mW under full outdoor irradiance. Fig. 5 shows the I-V and P-V curves under different irradiance levels.

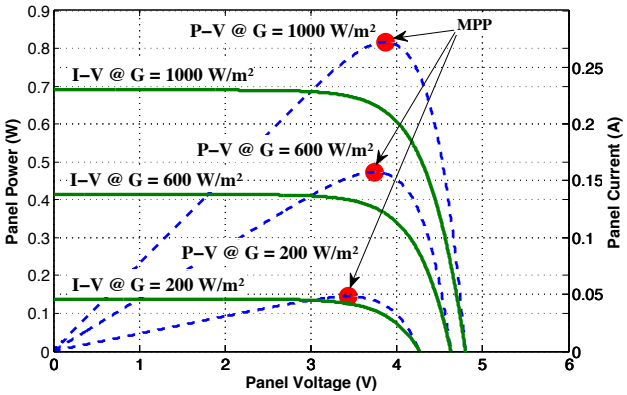

Fig. 5. PV panel current-voltage and power-voltage characteristic.

Since the available solar energy varies with time of day, a maximum power point (MPP) tracking mechanism is essential to take full advantage of available solar energy by continuously delivering the highest possible power to the battery or to the node. As shown in Fig. 5, the I-V characteristic of the PV panel is nonlinear and changes with irradiance.

The MPP tracker block consists of two parts: a DC-DC buck-boost converter and an MPP tracking (MPPT) control algorithm. The function of the DC/DC converter is to provide appropriate power to the load (battery or node) and compute the MPP according to the control algorithm. The converter input resistance R_{in}, which is the load resistance presented to the PV panel, varies according to the duty cycle D of the pulse width modulation (PWM) signal delivered by the MPPT control block and is therefore given as [9]

$$R_{\text{in}} = \left(\frac{1-D}{D}\right)^2 \frac{V_{\text{Load}}}{I_{\text{Load}}} \tag{1}$$

where V_{Load} and I_{Load} are respectively the converter output current and voltage. The MPPT controller is implemented using the widely used algorithm Perturb and Observe (P&O) [10]. This algorithm measures the power generated by the PV panel and finds and tracks its maximal value by varying the converter duty cycle.

B. Battery Model

There exist several models in the literature that capture battery charge/discharge behavior with reasonable reliability. For its simplicity and accuracy, we consider only the electrical circuit model based on previous work done by [11]. This model can predict both the $I-V$ performance and the lifetime and includes two important battery properties: the capacity fading effect due to charge-discharge cycles and temperature. The equivalent circuit model of the battery is represented in Fig. 6, where V_{oc}, V_{battery} and I_{battery} are the output open circuit voltage, the output voltage and the current of the battery, respectively.

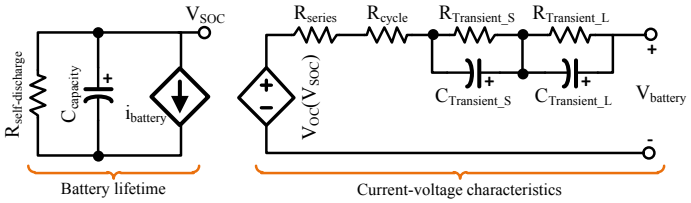

Fig. 6. The electrical battery model.

The battery used in the application below (*TCL PL-383562*) has a capacity of 850 mAh, with 400 cycles of charge and discharge. The expression of battery state of charge (SOC) is described by

$$\text{SOC} = \text{SOC}_{\text{init}} - \int \frac{I_{\text{battery}}}{C_{\text{usable}}}\, dt \tag{2}$$

where the SOC_{init} is the initial battery SOC and C_{usable} the usable battery capacity which is given as

$$C_{\text{usable}} = C_{\text{init}}.(1 - \text{Calendar life losses} - \text{Cycle life losses}) \tag{3}$$

where C_{init} is the initial battery capacity, *Calendar life losses* is the storage losses occuring when battery is not used and *Cycle life losses* is the capacity fading due to cycle number.

C. Power Manager

The power manager adopted is based on an ideal controller which manages the energy harvested and node power supply while ensuring that the battery is not overcharged nor over-discharged ($10\% \leq \text{SOC} \leq 95\%$). Additionally, this power manager periodically indicates to WSNet the energy source used, either solar harvester or battery. This information allows the sensor node to adapt its operations to the available power supply. The flowchart of the power manager is shown in Fig. 7.

Here, $V_{\text{Thr1}} = 3.6V$ and $V_{\text{Thr2}} = 4V$ are calculated according to the operating voltage of the sensor node and the battery. At reception of a synchronization or communication event from WSNet, the power manager measures the voltage delivered by the solar harvester. This voltage and the battery terminal voltage are compared to the configured thresholds V_{Thr1} and V_{Thr2}, the power manager determines which power source, either solar harvester or battery, should power the node. If the scavenged voltage is higher than V_{Thr2}, the power manager connects the solar harvester to the node and charges the battery only if the battery SOC is below 95%. Otherwise,

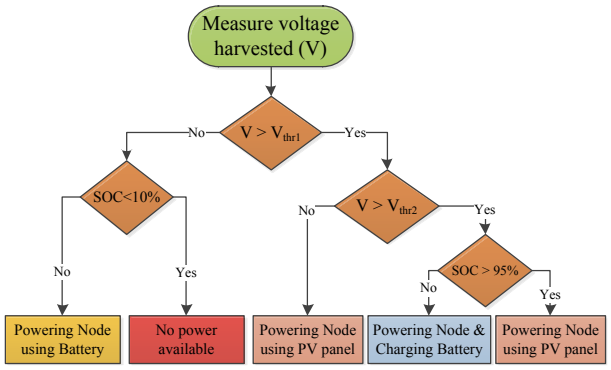

Fig. 7. Flowchart of the power manager.

if the PV panel voltage is lower than V_{Thr2} but higher than V_{Thr1}, the power manager connects the node directly to the PV panel. In the case where the voltage is below V_{Thr1}, the node is powered using the battery. Fig. 8 shows the evolution of the solar harvester and battery voltages during 4 days, from July 1 to July 4, as function of the solar irradiance data of Grenoble given by [12].

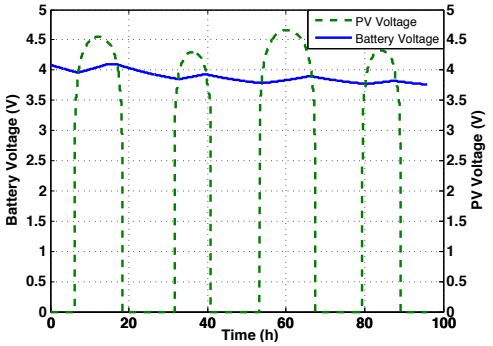

Fig. 8. Evolution of voltage in the energy harvesting system over 4 days.

V. A Case Study: Greenhouse Temperature Monitoring

In this section, we study the lifetime of a WSN for a temperature monitoring scenario located in a greenhouse using HarvWSNet. The simulated network consists of 8 sensor nodes distributed in a circular area around a base station as illustrated in Fig. 9.

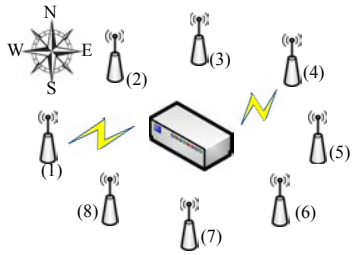

Fig. 9. Simulation set-up.

The communication distance between a sensor node and the base station is 40 m at a frequency of 2405 MHz and a

data rate of 250 kbps (IEEE 802.15.4). The MAC protocol used is CSMA/CA. All simulated sensor nodes are identical and have the following functionalities: a temperature sensor, a micro-controller for data processing, an RF transceiver to communicate with the base station, and the energy harvester described in Section IV for power.

We associate one year different irradiance data for each node depending on its position and solar panel orientation. The base station is assumed to have an unlimited power source. We consider the power consumption model of the board eZ430-RF2500 combining an RF transceiver CC2500, a micro-controller MSP430 and a temperature sensor [13]. The nodes are configured to sense and transmit the temperature at a 1 second transmission period to the base station. The average current consumed during sleep mode is given as

$$\begin{aligned} I_{\text{sleep}} &= (I_{\text{idle}}(\text{CC2500}) + I_{\text{idle}}(\text{MSP430})) \times (T_{\text{App}} - T_{\text{Tx}}) \\ &= (1.3[\mu A]) \times (1[s] - 0.002383[s]) \\ &= 1.296\ \mu A.s \end{aligned} \tag{4}$$

Therefore, the average current consumption of the application over 1 second is

$$\begin{aligned} I_{\text{avg}} &= (I_{\text{sleep}} + I_{\text{Tx}}(\text{MSP430}))\,/1[s] \\ &= (1.296[\mu A.s] + 35.508[\mu A.s])\,/1[s] \\ &= 36.80\ \mu A \end{aligned} \tag{5}$$

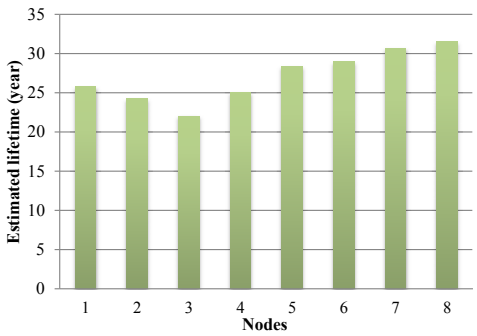

Fig. 10. Estimated lifetime of each sensor node.

Fig. 10 shows the estimated lifetime for each sensor node assuming that both the power manager and the MPP tracker consume 1 mW. We observe that certain nodes have longer lifetimes due to the solar data used which depends on solar panel orientation. The nodes facing south receive higher solar irradiance than nodes facing the north. Concerning battery lifetime, we have observed that most battery charge-discharge cycles are consumed during the winter months where the nodes do not receive sufficient solar irradiance. Even if the previous observations are predictable, HarvWSNet is the only simulation platform capable of producing accurate lifetime projections. Most importantly, for the same scenario described above, the processing time taken to simulate one year using WSNet with a simple linear battery model is about 3 hours, compared to 4,5 hours using HarvWSNet. This extra processing time is clearly worth the price since the evaluation of an energy harvesting node's lifetime is impossible using a linear battery model.

VI. Conclusion

While simulations are widely used to evaluate protocols and algorithms for WSNs, available network simulators offer only a simple linear battery model which can not be used to accurately estimate the lifetime of deployed energy harvesting nodes in the real world. In this paper, we have proposed HarvWSNet a co-simulation framework for energy harvesting wireless sensor networks. HarvWSNet is a global tool that combines the strengths of WSNet and Matlab to evaluate energy-aware protocols and hardware for WSNs. WSNet is used for the multi-node network simulation while MATLAB is used for the simulation of the energy harvesting system. Both tools are run interactively and synchronized using TCP sockets. To demonstrate the efficiency of HarvWSNet, a simulation case study based on a solar harvesting WSN is defined. To this end, a solar energy harvesting model is implemented in MATLAB based on COTS devices. A comparison of the runtime of HarvWSNet versus WSNet clearly demonstrates the efficiency of the former, even when a complex model of the energy harvesting subsystem is used. Our tool easily adapts to further refinements of the energy harvester model. The use of HarvWSNet facilitates research around power management strategies and power aware algorithms and hardware for the emerging class of perpetual energy harvesting wireless sensor applications. The HarvWSNet source code is available at amine.didioui@cea.fr.

Acknowledgments

This work was supported by the French Armaments Procurement Agency (DGA) and the French National Research Agency (ANR) project GRECO bearing reference ANR-2010-SEGI-004-04.

References

[1] MEMSIC. (2010) Mpr & mib user manual. [Online]. Available: http://memsic.com/

[2] J. Paradiso and T. Starner, "Energy scavenging for mobile and wireless electronics," *Pervasive Computing, IEEE*, vol. 4, no. 1, pp. 18 – 27, jan.-march 2005.

[3] A. Boulis. (2011) Castalia: A simulator for wireless sensor networks and body area networks. [Online]. Available: http://castalia.npc.nicta.com.au/

[4] G. Chelius, A. Fraboulet, and E. Benhamida, *WSNet - an Event-Driven Simulator for Wireless Networks*, http://wsnet.gforge.inria.fr/.

[5] Z. Zhang, Z. Lu, Q. Chen, X. Yan, and L.-R. Zheng, "Cosmo: Co-simulation with matlab and omnet++ for indoor wireless networks," in *Proc. IEEE Global Telecommunications Conf. GLOBECOM*, 2010.

[6] T. Kohtamaki, M. Pohjola, J. Brand, and L. M. Eriksson, "Piccsim toolchain - design, simulation and automatic implementation of wireless networked control systems," in *Proc. Int. Conf. Networking, Sensing and Control ICNSC '09*, 2009, pp. 49–54.

[7] M. S. Hasan, H. Yu, A. Carrington, and T. C. Yang, "Co-simulation of wireless networked control systems over mobile ad hoc network using simulink and opnet," *IET Communications*, 2009.

[8] E. Ben Hamida, G. Chelius, and J. M. Gorce, "Impact of the physical layer modeling on the accuracy and scalability of wireless network simulation," *Simulation*, 2009.

[9] E. Lefeuvre, D. Audigier, C. Richard, and D. Guyomar, "Buck-boost converter for sensorless power optimization of piezoelectric energy harvester," *IEEE Transactions on Power Electronics*, 2007.

[10] A.K.Mukerjee and N. Thakur, *Photovoltaic Systems: Analysis and Design*. Prentice-Hall, 2011.

[11] O. Erdinc, B. Vural, and M. Uzunoglu, "A dynamic lithium-ion battery model considering the effects of temperature and capacity fading," in *Proc. Int Clean Electrical Power Conf*, 2009.

[12] http://www.soda-is.com/.

[13] TI. Wireless sensor monitor using the ez430-rf2500. [Online]. Available: http://www.ti.com/litv/pdf/slaa378d