Back to Results Prototyping an Energy Harvesting Wireless Sensor Network Application Using HarvWSNet

Florian Broekaert, Amine Didioui, Carolynn Bernier, Olivier Sentieys

ARCS 2013, February 19 – 22, 2013, Prague, Czech Republic

Prototyping an Energy Harvesting Wireless Sensor Network Application Using HarvWSNet

Florian Broekaert, Thales Communications & Security, Gennevilliers, France
Amine Didioui, CEA, DRT, LETI, Grenoble, France
Carolynn Bernier, CEA, DRT, LETI, Grenoble, France
Olivier Sentieys, INRIA, IRISA, Lannion, France

Abstract

In this article, the HarvWSNet simulation framework is used to explore the feasibility of a structural health monitoring application based on a wireless sensor network. The aim is to show the value of a simulation-based approach for the architecture exploration and prototyping of severely energy constrained applications. In the target application, each network node performs a reading of temperature and/or acceleration and transmits the data to a base station every given period. A second objective of this work is to test the relevance of a power management algorithm whose role is to adapt the application quality of service to the instantaneous state of the energy harvesting module. The final goal is to propose a '*perpetually powered'* node architecture compatible with the user application requirements.

1 Introduction

Wireless sensor networks (WSN) are commonly used in applications such as environmental monitoring, structural health monitoring, healthcare, smart buildings, etc. However, since most of today's WSN's are battery-based systems, the application lifetime is either inherently limited by the battery capacity or extra costs are incurred when batteries must be replaced or recharged. Battery replacement may even be difficult or impossible in inaccessible or hostile environments. To avoid these limitations, recent advances in energy harvesting technologies allow nodes to extend their lifetime by scavenging energy (solar, thermal, mechanical, etc.) from the environment.
However, as opposed to a battery, the energy delivered by these sources is limited, non-constant and highly dependent on a variety of environmental factors. The success of a WSN application based on energy harvesting therefore depends on simultaneously mastering the power/performance trade-off of each component of the network as well as the impact of each component on the power/performance trade-off of every other component. A successful application is therefore the result of a complex multi-parameter optimisation.

1.1 Simulation-based architecture exploration and prototyping

The deployment of novel applications based on Energy Harvesting Wireless Sensor Networks (EH-WSN) depends on the development of dedicated components. Components of critical importance include wireless routing and medium access protocols, low power data processing units, RF transceivers, and energy management (harvesting, storage, conversion) modules. A power optimisation of each of these components is essential to the success of an energy constrained system.
To evaluate the feasibility of an applicative scenario, all of these components must be validated simultaneously. To this end, 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.
An alternative approach that avoids these difficulties is to prototype the applicative scenario using a wireless sensor network simulator. Of course, the validity of the observations that will be extracted from the simulation results directly depends on the comprehensiveness and accuracy of the simulation model of the different components. The calibration of each model component with respect to real-world measurements is therefore an essential task in this approach.

1.2 The Power Manager Approach

Another essential component of an EH-WSN is the power manager whose task consists in adapting the application to the time-varying energy source. The challenge is to adapt and balance the energy consumed by the system to the actual (and also future) supplied energy ensuring that the system can be completely autonomous in energy. In other words, the role of the power manager is not only to reduce the node's energy footprint by applying power optimisation techniques but it also has to prevent the system from over-consuming energy. The challenge is making sure the node will function as long as possible with a performance level that complies as much as possible with the user requirements. As shown in **Fig. 1**, the power manager is the central component of the system and has several interactions with the other elements of the system. In usual EH-WSN applications, the most common components are the sensors, the microcontroller, the RF transceiver, the battery and the energy harvester.

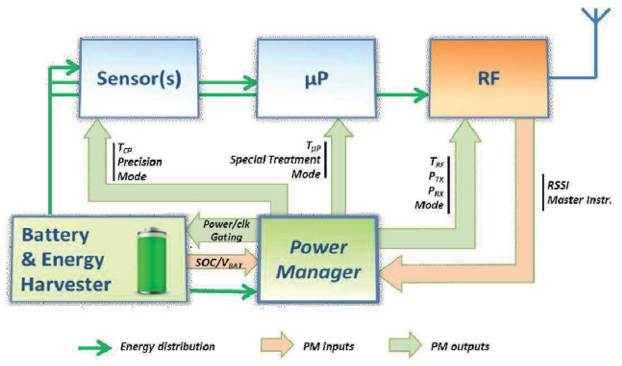

Fig.1 Power Manager within an EH-WSN node

End-user performance requirements are typically translated into specific Quality of Service (QoS) levels that the WSN application fulfils. By taking benefit of the HW scalability (thanks to HW knobs such as modes, duty cycle intervals, transmit power, etc), and the SW scalability (QoS degradation), the power manager can adapt, at runtime, the performance and the power consumption of the system. This scalability allows the power manager to meet performance requirements without wasting energy, by selecting an appropriate HW configuration for each application QoS level. Consequently, for each QoS level, there is a given power budget. Based on this information, power management strategies must be implemented taking into account internal events (e.g. remaining battery level, predicted amount of energy harvested, etc), and external events (e.g. received signal strength, master node instructions, etc).

2 The HarvWSNet simulation Framework

The HarvWSNet simulation Framework was presented in [1]. This framework **(Fig. 2)** was developed based on the observation that, since network and energy harvesting events are generally completely uncorrelated, better efficiency is obtained by combining the strengths of two different tools, namely MATLAB and WSNet, to produce realistic simulation results for the evaluation of EH-WSN performance and lifetime.

WSNet [2] is a discrete-event driven simulator perfectly adapted to the unevenly time-distributed nature of packet generation. It is therefore used to implement the communication protocol layers and simulate the network behaviour. Its specific strength over other network simulators is its detailed modelling of the radiofrequency communication medium [3].

The existing simple linear battery model of WSNet is replaced by a continuous time model of the energy harvesting system implemented in MATLAB. Indeed, an energy harvester is a dynamic system, where the battery state-of charge (SOC) and energy harvested change in a continuous manner with respect to time. Therefore, modelling 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 behaviour.

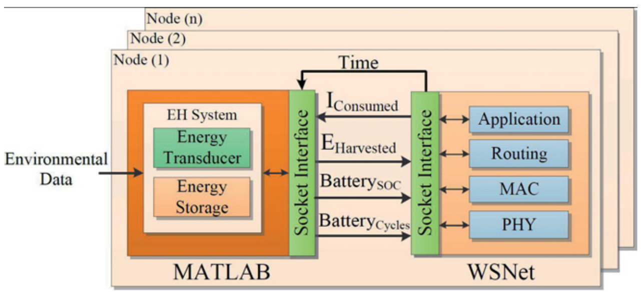

Fig. 2 The structure of HarvWSNet

In the HarvWSNet framework, both MATLAB and WSNet are run interactively and exchange data using sockets. WSNet is responsible for clock synchronization and simulation time handling. WSNet invokes Matlab at each *synchronization event*, scheduled at fixed intervals of time and dedicated to maintaining the synchronization of the simulation clock between WSNet and Matlab, as well as at each *communication event*, which arise when a node is in the following states: transmitting, receiving, listening, and sleeping **(Fig. 3)**. WSNet sends a notification to Matlab that includes the current to draw as well as the current draw duration. It then waits until the end of the Matlab calculation which potentially returns information concerning the energy harvesting module, e.g. the battery voltage or state-of-charge (SOC), the state of the module (charge, discharge, etc.), the instantaneous irradiance for a photovoltaic harvester, etc.

The energy harvester system of HarvWSNet is completely modular and can adapt to any type of harvested energy from simple to more complex models. Using HarvWSNet, it is possible to simulate the network behaviour 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.

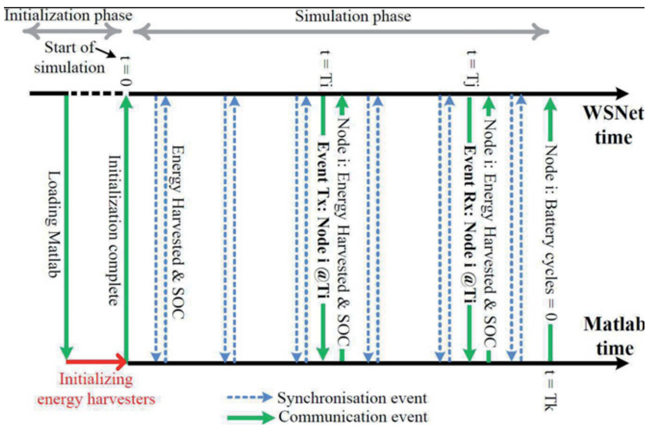

Fig. 3 Interactive co-simulation of WSNet and Matlab

3 The HUMS application

The features provided by the HarvWSNet framework have been evaluated in our trials to simulate a Health Usage and Monitoring System (HUMS) application. As depicted in **Fig. 4**, HUMS consists in lightweight and low-power systems with wireless capacities that are increasingly deployed by industrials on their equipments to monitor the operating constraints (humidity, pressure, temperature, vibrations…) that the equipment will undergo during its operational life. Depending on the system architecture, two main usage profiles are commonly seen:

- Periodically (daily, weekly, monthly) stored collected data are sent for further analysis to a coordinator station or retrieved manually on the equipment by an operator.
- If a collected value is out of a predefined range, an alert message has to be sent as soon as possible to the coordinator station.

For both usages, the two most critical functional requirements that HUMS have to comply with are to miss no critical events and to send an alert message as soon as a collected value has over passed a predefined threshold.

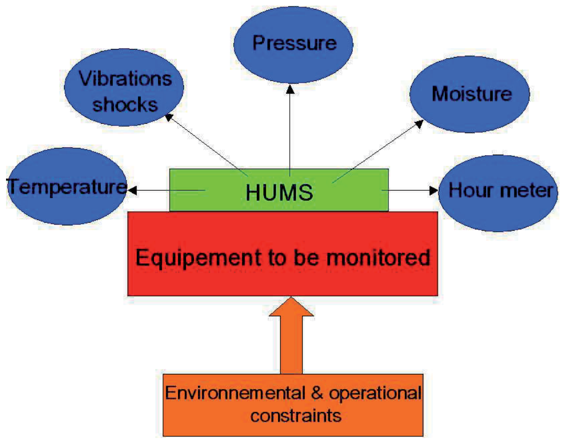
Pressure
Vibrations shocks
Moisture
Temperature
HUMS
Hour meter
Equipement to be monitored
Environnemental & operational constraints

Fig.4 Example of a HUMS application

These collected data are analysed to readjust the Mean Time Before Failure (MTBF) statistics. These HUMS are especially useful for Service and Support teams to collect return on experience information and reduce costs generated by useless preventive maintenance actions.

For this application, energy harvesting constitutes an opportunity to replace the traditional coin cells batteries used to power these sensors. Batteries are spare part that requires preventive maintenance and which are typically replaced every 18 months (this period can vary depending on the use case).

As described in section 1.2, the use of energy harvesters for HUMS requires the development of a smart power manager. This module must ensure that the system accomplishes its missions even with a limited energy source that may not always supply energy.

In our study, the prototyping platform used for the experiments is the TI-EZ430-SEH based on a MSP430 microcontroller and a CC2500 transceiver. To realize the HUMS functionality, we have defined a SW architecture based on three main functions **(Fig. 5)**

- **The SCU module**: dedicated to sensor handling, data processing, threshold comparison and storage. In our trial a temperature sensor and a three axis accelerometer sensor are used.
- **The RFT module:** dedicated to the radio communication from packet elaboration to transmission of the samples to the antenna. In our trial a 802.15.4 protocol similar to Zigbee is used for the wireless communication.
- **The PMS module:** dedicated to the configuration of the different component power/performance modes and the power source and storage handling.

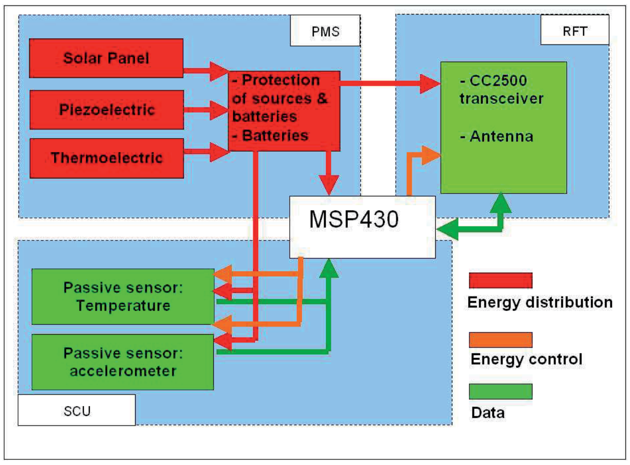

Fig. 5 Internal structure of the HUMS system

In this architecture, the PMS corresponds to the power manager. Its task is to decide what amount of power can be distributed to the system depending on the remaining energy available in the battery and the amount of energy harvested. Quality of Service (QoS) levels are defined which identify predefined modes (corresponding to a set of performance and power consumption levels) resulting in the adaptation of both the application and the hardware configuration. For our HUMS application, three QoS levels have been defined (low, medium and high) corresponding to different modes for the three modules SCU, PMS, RFT.

For the SCU module, the parameters which can be modified are the processing precision (e.g. 8, 16 bits), the quantity of collected samples (i.e. sampling frequency), the periodicity of the sensing as well as the level of post sensing processing to be applied (mean, comparison to threshold values). For the RFT, parameters such as the transmit power, the number of bits transmitted as well as the periodicity of the transmissions can be configured. For the PMS module, the interval between two wake-ups can be configured.

The **Table I** details all the different modes for the various QoS levels.

	QoS High	QoS Medium	QoS Low
PMS periodicity	5sec to 10sec	20sec to 30sec	30sec to 60sec
SCU periodicity	2sec to 10sec	20sec to 60sec	120sec to OFF
SCU sensing types	T°c & accelero	T°c & accelero	T°c or none
Acquisition granularity	Clock x8 for T°C 3.2KHz for accelero	Clock x8 for T°C 100Hz for accelero	Clock x8 for T°C
Processing precision	16 bits	16 or 8 bits	8 bits
SCU post process-ing	Mean, maximum, minimum, history	Mean, History	History
RFT periodicity	3sec to 20sec	25sec to 111sec	OFF
RFT transmission power	Max	Min	Min

Table I PMS/SCU/RFT modes for each QoS level

For each Quality of Service level, PMS, SCU and RFT are actually divided into eight sub-modes:

- High : modes from 5 to 7
- Medium : modes from 2 to 4
- Low : modes from 0 to 1

The power management strategy that is implemented in the PMS relies on the percentage of the remaining battery capacity and the harvesting state (charge/discharge). To evaluate the remaining battery capacity, a state of charge monitor is not available on the HW platform prototype. Consequently, input voltage is used instead, even if this parameter is not as good as the state of charge to provide a very accurate view of the remaining battery capacity. When the PMS function is executed, it checks the battery level and reconfigures the PMS, SCU and RFT modes according to the following **Table II**.

QoS level		Low	Low	Low	Medium	Medium	Medium	Medium	High	High
Battery voltage (V)		Min	2,38	2,63	2,83	2,98	3,10	3,20	3,28	Max
PMS	Charge	Mode 0	0	1	1	1	7	6	6	
	Discharge	0	0	1	2	3	4	5	6	
SCU	Charge	Mode 0	0	1	1	1	7	6	6	
	Discharge	0	0	1	2	3	4	5	6	
RFT	Charge	Mode 0	0	1	1	1	7	6	6	
	Discharge	0	0	1	2	3	4	5	6	

Table II PMS/SCU/RFT modes table

The modes for the different battery capacity thresholds are set experimentally. One of the objectives of the HarvWSNet simulator will be to refine this table based on the simulation results.

4 Simulation model calibration

As stated above, our prototype network node is based on the popular TI-EZ430-SEH platform. An essential part of the pre-simulation study consists in measuring the actual current drain of the platform in its different modes and calibrating the corresponding simulation models. Indeed, it was found that the measured current drain is consistently superior to the values specified in the datasheets.

The TI-EZ430-SEH [4] platform includes an MSP430 microcontroller with 128kbyte internal memory coupled to a CC2500 RF transceiver. The SimpliciTI [5] network protocol stack is used in our trials. The board can be connected to a solar energy harvesting module containing a Cymbet 2.5”x2.5” solar panel (that provides 350µAh under 1000 Lux conditions) and two EnerChip CBC050 batteries for a total useful capacity of 65µAh. A TMP122 temperature sensor is present on the platform to which we have also added a 3 axis ADXL345 accelerometer.

Our first task consisted in measuring the current drain of the platform by measuring the voltage drop over a 1 Ohm resistance placed in series with the voltage supply. Measurements were made using a Tektronix MSO 4104 oscilloscope, a Fluke 23 multimeter and an HP 3478A voltmeter. The objective was to characterize the current drain in the modes defined above: PMS, SCU, RFT and sleep. The current draw corresponding to these events is shown on **Fig 6**. Pauses are placed in the application code in order to easily identify the current drain for each mode.

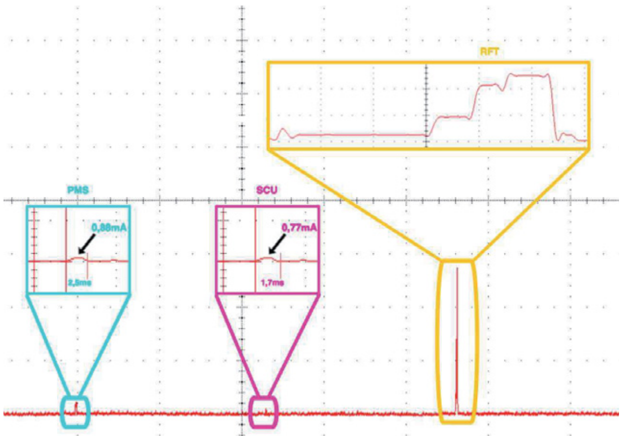

Fig. 6 Current trace for PMS, SCU and RFT modes

In this example, the PMS current drain corresponds to the platform’s current dissipation while the MSP430 samples the supply voltage using an internal ADC and calculates the new power manager settings. Current drain is approximately 0.88mA during 2.5msec, or 2.2µAsec.

Similarly, the SCU current drain represents the current dissipation required to sample the temperature sensor using an ADC on the MSP430 and to perform sensor data filtering, data history management (if needed), and maximum, minimum, and average value calculations. Current drain is approximately 0.7mA during 1.7msec, or 1.2µAsec. If the accelerometer is also read (assuming a

100Hz sampling rate), the SCU current drain is 1.36mA during 19.76msec, or 26.8µAsec.

The RFT current drain corresponds to the energy required for packet assembly, frequency synthesizer settling, clear channel assessment (CCA) and transmission of a 7 useful byte packet at the default 0dBm emission power. If power is integrated over time, required energy is 46.4µAsec.

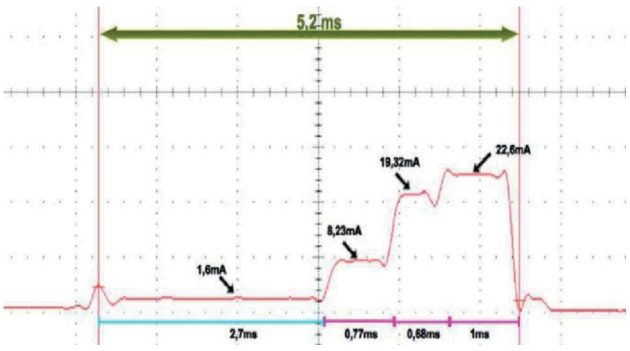

Fig. 7 Detailed current trace in RFT mode

Estimated (from datasheets) current drain of the platform in sleep mode is 1.4µA with respectively 0.9µA for the MSP430, 0.4µA for the CC2500 and 0.1µA for the ADXL. We configured the application in order to have an interval of one second between two cycles (one cycle consists in the sequential execution of the PMS+SCU+ RFT). As a result, every second the application spends 0.992s in sleep mode. The resulting power consumption is 1.39µAsec.

When application is configured this way (i.e. one cycle PMS+SCU+RFT every second), its associated power budget is 51.2µAsec (0.0142µAhour)

To verify that the simulation model was correctly calibrated, the platform's lifetime was measured with a PMS+SCU+RFT event as described above (without the accelerometer reading) performed every second. Starting with fully charged batteries and with the solar harvester disabled, the system has a measured lifetime of 1h22 minutes.

To model the platform using HarvWSNet (**Fig. 2**), mod els of the PMS and SCU are defined in the application layer. The RFT mode is modelled using the appropriate network, MAC and PHY protocol models. Any component of the protocol stack and application layer can initiate current discharge events that are handled by calls to a Matlab model of the energy harvester (here limited to a battery discharge model). After each call, the updated battery state of charge (SOC) and harvester state (charge/discharge) are returned to WSNet to serve as inputs to the power management algorithm. A SOC of 0 corresponds to the node's end of life.

To compare the simulation and the measurements, a simple two node network simulation is set up where an emitting node performs the previously defined PMS+SCU+RFT events every second. With these settings, the simulation lifetime of the emitting node is 1h22minutes and 46 seconds. This corresponds to a difference of 0.6% with respect to the measured results from which we conclude that the simulation models are correctly calibrated.

5 Examples of architecture exploration using HarvWSNet

Besides using HarvWSNet as an environment to evaluate different power management policies or to configure the power manager parameters, the features provided by this simulation framework can ease the validation of the key elements of a design. Indeed, today's system architects lack unified tools that can allow them to test the components involved in the different layers of a WSN with accuracy and fast simulation times.

Depending on the use case and the operational scenario, there are a number of key parameters (such as memory capacity, battery capacity, size and number of packets to be transmitted, energy harvester size and efficiency, etc…) that may introduce bottlenecks in the architecture and require proper dimensioning. Using HarvWSNet, these parameters can be easily modified in order to evaluate their impact on the system behaviour and autonomy.

For example, soda-is [6] provides detailed irradiance profiles of various cities. This information can be injected into the Matlab part of the HarvWSNet simulator to define a daily irradiance profile corresponding to a city location close to the operational context studied (**Fig. 8**).

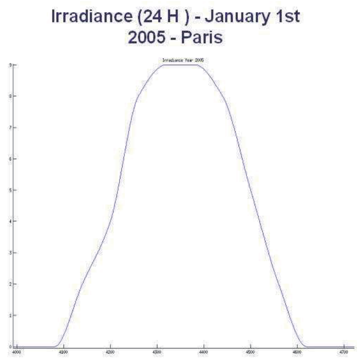

Fig. 8 One day irradiance profile in Paris

Changing this irradiance profile is easy and, as shown in **Fig. 9**, it is also possible to generate a profile covering longer periods of time.

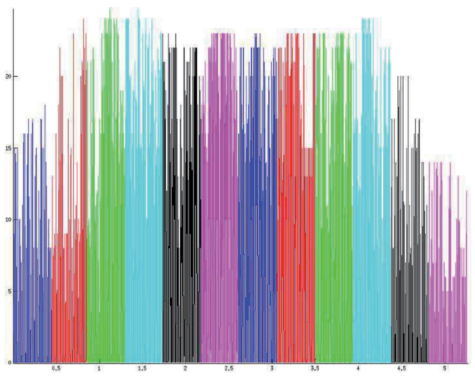

Fig. 9 One year irradiance profile in Paris

Fig. 10 and **Fig. 11** show a simulation of the HUMS application with a PMS+SCU+RFT cycle every second. In **Fig. 10** we wanted to highlight the impact of modifying the number of cells of the solar energy harvester.

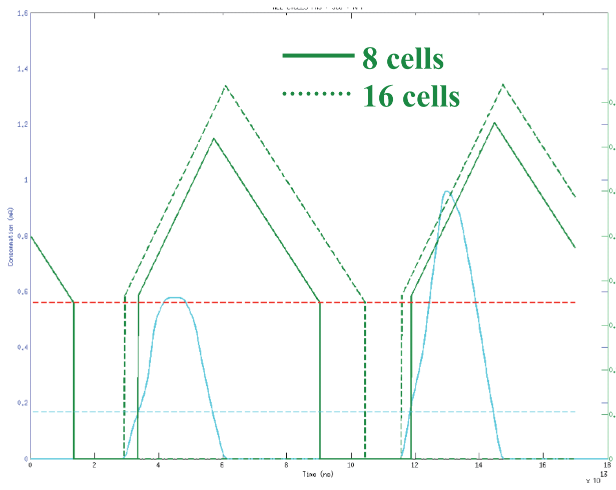

Fig. 10 Simulation with various numbers of cells in a solar energy harvester

In this figure, the green curves represent the battery capacity (100µAhour in this experiment) level normalized to 1. From that curve the autonomy can be observed. The blue curve represents the irradiance profile during one day. The blue dashed curve corresponds to the minimum irradiance level required by the energy harvester to supply energy and the red dashed curve represents the battery capacity level corresponding to the cut-off threshold. The green curve shows that with an eight cell solar energy harvester, the battery is recharged to a certain level (not fully charged), whereas with a sixteen cell harvester (green dashed curve), the battery is recharged to a higher level. Note that, in this experiment, even if the solar panel provides twice the current, the battery charging current is limited by the maximum charge current of the battery.
The **Fig. 11** shows a second experiment where batteries with various capacities have been tested.

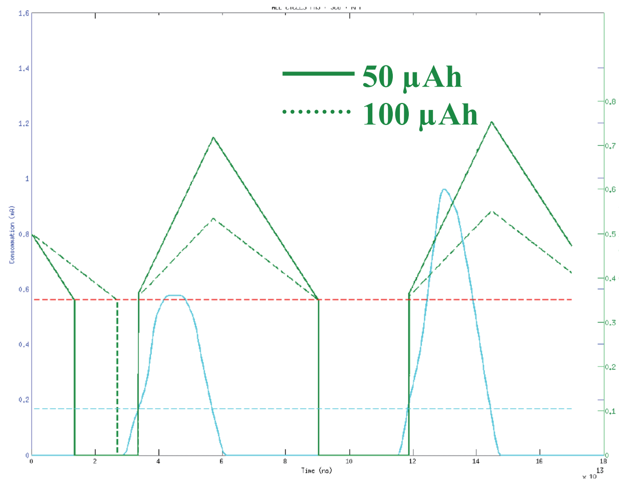

Fig. 11 Simulation with various battery capacities

The green curves represent the battery capacity level (normalized to 1). The green curve corresponds to a battery of 50µAhour and the green dashed curve corresponds to a battery of 100µAhour. At the beginning of the simulation both batteries are charged at 50%. We observe that, in this irradiance scenario, doubling the battery capacity does not increase the system autonomy.
Thanks to the large library of models available with this simulator and its fast simulation time, HarvWSNet is useful to support design space exploration. In addition, various other operational conditions can be tested by modifying system level parameters such as distance between nodes, communication channel, battery technology, protocol, harvesting sources, etc.

6 Conclusion

In this work, we have investigated the use of the HarvWSNet framework for the exploration of the prototyping space of a sensor network application. To this end, simulation models of the different components of the node, including the energy storage module, have been developed and calibrated versus measurements. The use of such a framework not only eases the development of new applications, it can also be used to test non existent hardware and innovative protocols and algorithms. One such algorithm is the power manager whose task is to adapt the quality of service provided by each component of the node to the instantaneous state of the energy harvesting module. Our results have demonstrated the value of using such a simulation framework for the analysis of the multi-dimensional optimisation problem posed by energy harvesting wireless sensor networks.

7 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.

8 References

[1] A. Didioui, C. Bernier, D. Morche, O. Sentieys, “HarvWSNet: A Co-Simulation Framework for Energy Harvesting Wireless Sensor Networks”,IEEE ICNC, January 2013.

[2] Chelius, G.; Fraboulet, A.; Ben Hamida, E.: WSNet -an Event-Driven Simulator for Wireless Networks, http://wsnet.gforge.inria.fr/

[3] Ben Hamida, E.; Chelius, G.; Gorce, J.-M.: Scalable versus Accurate Physical Layer Modeling in Wireless Network Simulations, 22nd Workshop on Principles of Advanced and Distributed Simulation (PADS), 2008

[4] http://www.ti.com/tool/ez430-rf2500-seh

[5] http://www.ti.com/lit/ml/swru130b/swru130b.pdf

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