Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 136745, 11 pages
http://dx.doi.org/10.1155/2013/136745

Research Article

Hardware Architecture Design for WSN Runtime Extension

Ángel Asensio, Rubén Blasco, Álvaro Marco, and Roberto Casas

Institute of Engineering Research (I3A) of the University of Zaragoza, 50018 Zaragoza, Spain

Correspondence should be addressed to Roberto Casas; rcasas@unizar.es

Received 3 August 2012; Accepted 1 April 2013

Academic Editor: Ling Wang

Internet of Things imposes demanding requirements on wireless sensor networks as key players in context awareness procurement. Temporal and spatial ubiquities are one of the essential features that meet technology boundaries in terms of energy management. Limited energy availability makes anywhere and anytime sensing a challenging task that forces sensor nodes to wisely use every bit of available power. One of the earliest and most determining decisions in the electronic design stage is the choice of the silicon building blocks that will conform hardware architecture. Designers have to choose between dual architectures (based on a low-power microcontroller controlling a radio module) and single architectures (based on a system on chip). This decision, together with finite state machine design and application firmware, is crucial to minimize power consumption while maintaining expected sensor node performance. This paper provides keys for energy analysis of wireless sensor node architecture according to the specific requirements of any application. It thoroughly analyzes pros and cons of dual and single architectures providing designers with the basis to select the most efficient for each application. It also provides helpful considerations for optimal sensing-system design, analyzing how different strategies for sensor measuring and data exchanging affect node energy consumption.

1. Introduction

Internet of Things (IoT) applications and scenarios are very heterogeneous: environmental monitoring in large areas [1], people monitoring in their own homes [2], or industrial environments [3] are some examples. This derives different requirements regarding network architecture and sensing nodes design [4]. According to Merriam-Webster dictionary, ubiquity is defined as the capacity of presence everywhere and in many places simultaneously. Sensors are today needed in different scenarios, and in all of them it is desirable that they be operative everywhere and every time they are required; for this reason, it is said that future sensors must be ubiquitous. It has two faces: spatial ubiquity—which inherently forces wireless communications and absence of wired power sources—and temporal ubiquity—which implies availability along functioning time (maximum energy autonomy) and also availability at any given time. Whichever the case, it leads to the common need of installation's runtime maximization and consequently minimization of energy demanded by sensing nodes [5]. There are many options to power wireless sensor nodes [6], but a real installation usually poses severe limitations: there is not unlimited power source available, energy from the environment is scarce and not enough for continuous running (e.g., indoors), maintenance of sensors is problematic (e.g., physically hard to reach to change batteries or expensive), and so forth. Thus, is critical to minimize node's power consumption while maintaining application's required quality of service. It is well known that power consumption has a high impact over quality of service offer by a WSN and its lifetime [3–5, 7]; the paper is centered on its analysis.

Depending on the deployment scenario, sensor duties will vary: data sensing, processing, aggregation, forwarding, sending, and so forth. In this paper we focus on a common case in many IoT applications: a sensor node periodically samples (every t_{SAMPLE}) one or more sensors (temperature, humidity, light, presence, chemical concentration, etc.), and then it performs some data processing and reports the readings to the network every t_{REPORT}.

Standard IEEE 1451 describes a set of open, common, network-independent communication interfaces for connecting transducers (sensors or actuators) to microprocessors, instrumentation systems, and control/field networks [8].

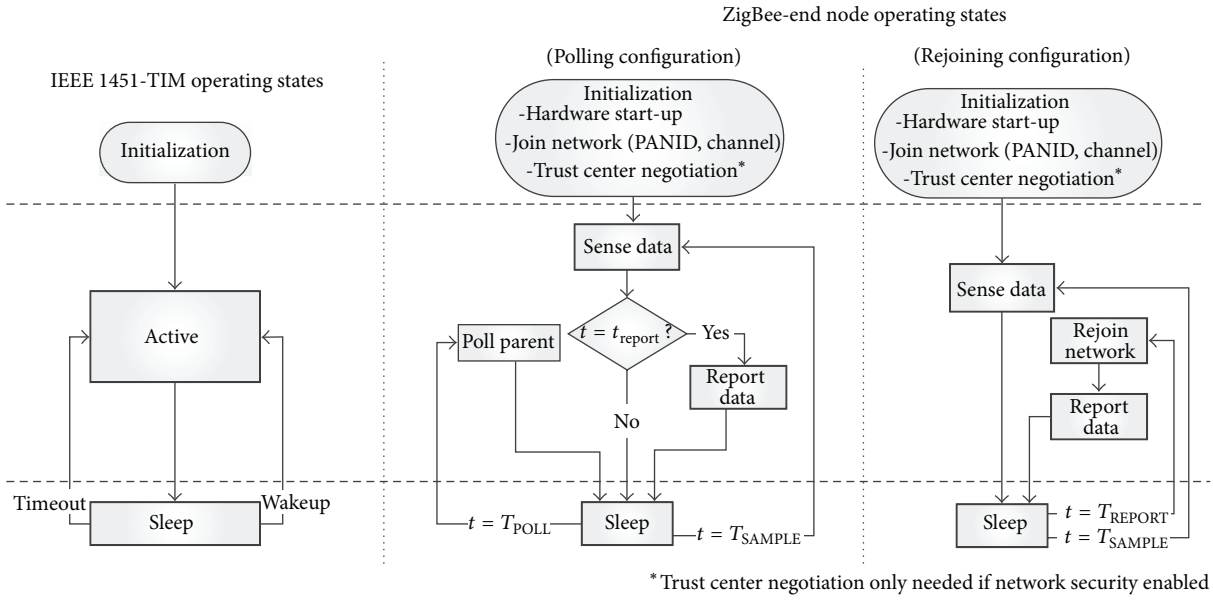

Figure 1: Finite state machine that defines a sensing node attending to IEEE 1451 and ZigBee standards.

IEEE 1451 introduces the concept of a transducer interface module (TIM) as a module that contains the interface, signal conditioning, analog-to-digital and/or digital-to-analog conversion, and, in many cases, the transducer. The specification defines a generic finite state machine (FSM) in Figure 1 that describes the operation of sensing nodes—TIMs—with three different operational states: initialization, active, and sleep [9].

IEEE 1451 is not restricted to any communication technology, and thus FSM definition is generic and leaves to each standard the specification of the substates needed. There are many WSN protocols available [10], and we select ZigBee for the study as it is a mature wireless standard for sensor networks, worldwide accepted, and with many hardware manufacturers available. The methodology described could be easily applied to any other standard. According to the standard specification [11], FSM states are defined as follows (Figure 1).

(1) *Initialization State*. Besides hardware startup (oscillator warmup, peripheral initialization, etc.), the ZigBee node has to initialize the network which means to check its network parameters (PANID—personal area network identifier—and channel mask), and if previously not joined to any network then scan the radio channels to discover available networks, join to a specific network, announce itself in the network,and, if the network has security enabled, wait to be authenticated by the Trust Center and for successful acquisition of the network key.

(2) *Active State*. Minimum tasks defined are polling its parent (to check if there are messages pending for the node), responding to any device discovery or service discovery operations requested, periodically requesting the Trust Center to update its network key (if security is enabled), processing device announce messages from other nodes, rejoining the network if disconnected for any reason, searching for alternative parent in order to optimize recovery latency and reliability, and so forth. Besides these network tasks, the node will also manage the sensors it might has, process and send sensor data, and so forth.

(3) *Sleep State*. It generically does not have any network or sensor and process duty assigned. This state is devoted to power electronics down to the maximum and to wait until there is any task to do switching to active state.

Temporal ubiquity of a wireless sensor node might suppose that communication with node must be guaranteed with a minimal latency time. This is commonly implemented following two different strategies that ensure lowest power of a wireless node: stay connected doing periodical network polls to receive incoming messages or leave the network and periodically reconnect. According to ZigBee specification, this is implemented following two different strategies shown in Figure 1.

(i) Polling configuration indicates that sensor node never leaves the network and periodically polls its "parent" (another node in the network that holds its messages while it sleeps).

(ii) Rejoining configuration indicates that sensor node leaves the network between reporting periods.

Both strategies are considered in ZigBee standard but no one is always more convenient than the other; while the first strategy guarantees that the node will receive messages from the network every time it polls, the second strategy reduces radio power consumption between reports to the minimum.

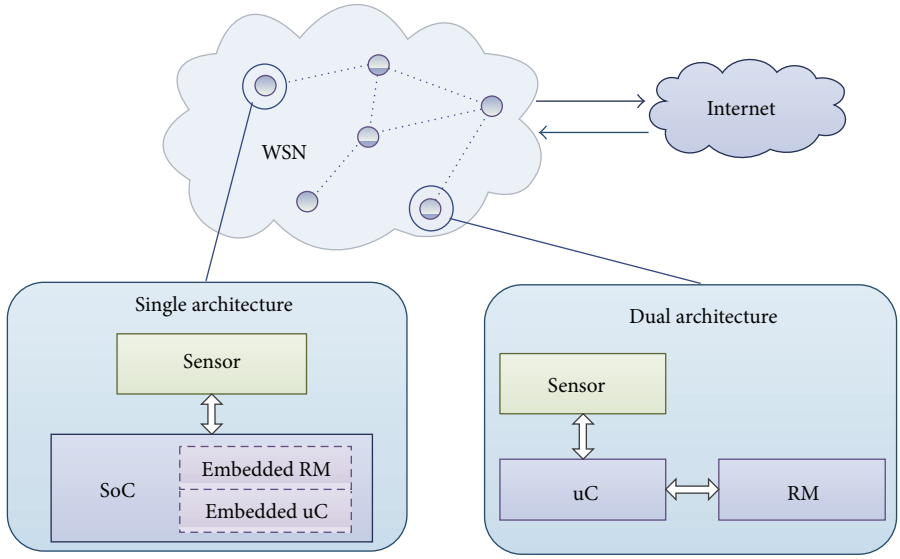

Figure 2: Sensor node's single and dual hardware architecture.

Energy required to retrieve and send data from the sensor to its destination must be as small as possible and its optimization needs from a multidisciplinary knowledge are improved electronic stages, network management optimization, cooperative tasking, or other alternatives [12]. It should be approached from a combined perspective [13] that merges network, (spatial distribution of network nodes [14], medium access control [15], routing [16], etc.) and node design considerations. Hardware [17] and firmware [18] design of the sensing node is crucial and it is usually done in a superficial way, just looking at the power requirements of the different hardware blocks and optimizing firmware [19].

This paper analyses energy issues associated with the different design alternatives. The next section shows main hardware architectures used to build a wireless sensor: single and dual. Then, based on the implementation of the previously described finite state machine, a mathematical model of energy consumption is defined. The energetic impact derived from hardware architecture and runtime pattern is presented in Section 4. Finally, several considerations about how design strategies impact over energy consumption and performance comparison of different WSN platforms are shown.

2. Hardware Architecture

The building blocks of a sensor node are power management, sensor, communication, and control and/or processing. Wireless communications are the power hungriest part in a node [20]; nevertheless, its impact in overall energy demand can be reduced as these systems optimize its use to the maximum. On the contrary, power consumption of the sensor is often lower compared to communications, but it can have larger influence on the overall system performance depending on how the node performs the measuring process (sampling rate, signal conditioning, data acquisition, etc.) [21]. As a consequence, hardware architecture of node is critical when implementing a real application and electronic designers must decide between two different architectures.

(1) Dual architecture is composed of a microcontroller (uC) that runs the application and control and a radio module (RM) that implements wireless communication. Depending on the radio module, it can just be a transceiver implementing the lowest ISO/OSI layers of a standard (e.g., TI's CC2420 [22], that is, IEEE 802.15.4 compliant) or implementing a specific wireless standard to the application level (e.g., Ember's EM260 network coprocessor implementing ZigBee stack). Both cases share in common the RM that is not programmed, but is just configured or controlled through Universal Asynchronous Receiver/Transmitter (UART), Serial Peripheral Interface (SPI), or Inter-Integrated Circuit (I^2C) protocols [23] using a set of commands provided by the manufacturer.

(2) Single architecture is composed of a system-on-chip (SoC) embedding a radio module and a programmable microcontroller. In this case, the hardware manufacturer provides wireless standard compliance through an API and/or development environment that the programmer uses and implements the application and downloads it to the SoC. (e.g., Ember's 35x with EmberZNet Pro [24] or TI's CC2530 [25] with Z-Stack).

As seen in Figure 2, both architectures can be used to implement a low power consumption end node. Hardware manufacturers are clearly pointing to single architectures in order to maximize energy efficiency, reduce complexity, easily design, and so forth. Nevertheless, is this always true?, under which conditions?, is the strategy of splitting tasks between two low power microcontrollers more convenient in terms of energy efficiency? [26]. In order to answer

Table 1: Power modes per silicon blocks (uC, RM, and SoC).

	Power mode 0 (PM_0)	Power mode 1 (PM_1)	Power mode 2 (PM_2)	Power mode 3 (PM_3)
Microcontroller	Deep sleep (low power timer running)	Low power (slow oscillator, peripherals interrupts on)	High power (fast oscillator)	Not applicable
Radio module	Deep sleep or powered off	Not applicable	Not applicable	Radio on
System on chip	Deep sleep (low power timer running)	Not applicable	Internal uC high power and radio off	Internal uC active and radio on

these questions, in the following sections we compare both architectures analyzing the energy consumption related to each state first theoretically (Section 3) and then with two real implementations.

3. Runtime Energy Consumption Analysis

Energy monitoring during design and commissioning of a wireless sensor network is challenging. Real measurement in specific nodes is possible [27]; nevertheless, WSN characteristics make it difficult to set nodal energy meters all over the network. Thus, it is common to use tools based on nodes' and networks' models that simulate hardware [28], data traffic [29], and associated energy consumption. As it is of key importance to understand the origin of every nanoamp in order to achieve the lowest power consumption [30] and due to the fact that there are no models that consider the architectures described, in the following we study in depth the energy associated with each substate and transition of the sensing node's FSM described in Figure 1.

Minimization of energy consumption is a tradeoff between strategy chosen, application times between events (t_{SAMPLE}, t_{REPORT}, and t_{POLL}), and hardware architecture. According to this, many authors propose different energy models, most of them differentiating between four silicon modules: microprocessor, transceiver, sensor, and power supply [31]. In this study, as we aim to compare the hardware architectures discussed in previous section, it is not needed to consider sensor and power supply models because both will equally affect the energy balance; for example, whichever sensor(s) we use, they will output a digital serial communication interface (e.g., SPI and I^2C) or an analog signal that will be, respectively, digitalized by the uC or the SoC.

The estimation of the power consumption of a sensor node is normally based on determination of each of the operation modes of the sensor [32]. These modes are highly influenced by the communication protocol and system hardware. In Table 1, we specify all the power modes in which a node will work.

Table 2 specifies the power mode in which the hardware (uC, RM, and SoC) of the sensor will be in order to work according to poll configuration scheme in Figure 1. (We use poll configuration as it is the most complex scenario and rejoin configuration eliminates "poll parent" state, and the PM_0 of the RM will be reduced, while $PM_{0\to x}$ will increase.) Energy necessary to switch between power modes is not negligible, especially when going from low power to high power [33], thus it is also indicated in Table 2.

Energy consumed in a given state "x" will be the sum of its "m" substates calculated as

$$E_x = V \times \sum_{j=0}^{m} \int_0^{t_j} i_j(t)\, dt = V \times \sum_{j=0}^{m} Q_j, \tag{1}$$

where V is the voltage supply and the second term is the integral of the current consumed i_j and during the time t_j the substate lasts.

Attending to the substates and considering the information that can be measured and extracted from hardware datasheets and application notes, the charge demanded by each state is defined in Table 3, where $I_{\text{UC,RM,SoC}_{0,1,2,3}}$ is the current consumed by uC, RM, and SoC in power modes 0, 1, 2, and 3 respectively, $Q_{\text{UC,RM,SoC}_{0,1,2,3}\to 0,1,2,3}$ is the charge drained by uC, RM, and SoC in transitions between corresponding power modes, $t_{\text{UC}_{0,1\to 1,2,3}}$ is the time needed by uC to change from modes 0 and 1 to 1 and 2, respectively, $Q_{\text{RM,SoC}_{\text{INIT,REPORT,POLL}}}$ is the charge drained by RM and SoC in network initialization, data report, and parent poll, $t_{\text{RM,SoC}_{\text{INIT,REJOIN,REPORT,POLL}}}$ is the time needed by RM and SoC in respective network process, t_{SENSOR} is the time needed by the sensing entities to sensor a valid measure in their outputs, $I_{\text{UC,SoC}_{\text{ACQ}}}$ is the current needed by uC and SOC for data acquisition from the sensing entities, for example, A/D conversion, $t_{\text{UC,SoC}_{\text{ACQ}}}$ is the time needed by uC and SOC for data acquisition from the sensing entities, for example, A/D conversion, $I_{\text{UC,RM}_{\text{SCI}}}$ is the current needed by uC and RM for data communication via serial communication interface, $t_{\text{SCI}_{\text{REPORT,POLL,POLL\_ANSW}}}$, is the times needed to communicate between RM and uC via serial communication interface, and t_{SLEEP} is the time in sleep mode.

As we aim to compare both architectures, many simplifications are possible.

(i) Terms related to network operations ($Q_{\text{RM,SoC}_{\text{INIT,SEND,POLL}}}$) and power state change ($Q_{\text{RM,SoC}_{0,1,2,3}\to 0,1,2,3}$) are equivalent in terms of energy consumption for RM and SoC. (This assumption can be considered as RM and SoC from the same manufacturer share the same radiofrequency hardware, for example, Texas Instruments' CC2520 transceiver and CC2530 SoC or Ember's EM357 coprocessor and EM357 SoC.)

(ii) Charge needed for network initialization is only consumed once and it is negligible compared to the charge needed by other states and consequently to the charge of the battery (below 0,05% with a 1000 mAh battery).

Table 2: Power modes in each substate of a normal operating cycle of a sensing node.

States	Substates	Dual architecture		Single architecture
		UC	RM	SoC
Init network	Scan channels	PM_0	PM_3	PM_3
	Discover networks	PM_0	PM_3	PM_3
	Join network	PM_0	PM_3	PM_3
	Announce node in network	PM_0	PM_3	PM_3
Sense data	Change power mode	$PM_{0\to1}$	PM_0	$PM_{0\to2}$
	Activate sensor and wait for data ready	PM_1	PM_0	PM_2
	Acquire data	PM_1	PM_0	PM_2
Report data	Change power mode	$PM_{0\to2}$	$PM_{0\to2}$	$PM_{0\to2}$
	Exchange "report_data" command (RM → uC)	PM_2	PM_3	—
	Change power mode	$PM_{2\to0}$	$PM_{2\to3}$	$PM_{2\to3}$
	Rejoin network (if not polling periodically)	PM_0	PM_3	PM_3
	Send data to the network	PM_0	PM_3	PM_3
Poll parent	Change power mode	$PM_{0\to2}$	$PM_{0\to2}$	$PM_{0\to2}$
	Exchange Poll event (uC → RM)	PM_2	PM_2	—
	Change power mode	$PM_{2\to1}$	$PM_{2\to3}$	$PM_{2\to3}$
	Poll parent in the network	PM_1	PM_3	PM_3
	Change power mode	$PM_{1\to2}$	$PM_{3\to2}$	—
	Exchange "poll response" (RM → uC)	PM_2	PM_2	—
Sleep	Change power mode	$PM_{x\to0}$	$PM_{x\to0}$	$PM_{x\to0}$
	Sleep	PM_0	PM_0	PM_0

Table 3: Consumption in each substate of a normal operating cycle of a sensing node.

States	Substates	Dual architecture	Single architecture
Init network	Scan channels Discover networks Join network Announce node in network	$I_{UC_0} \times t_{INIT} + Q_{RM_{INIT}}$	$Q_{SoC_{INIT}}$
Sense data	Change power mode	$Q_{UC_{0\to1}} + I_{RM_0} \times t_{UC_{0\to1}}$	$Q_{SoC_{0\to2}}$
	Activate sensor and wait for data ready	$(I_{UC_1} + I_{RM_0}) \times t_{SENSOR}$	$I_{SoC_2} \times t_{SENSOR}$
	Acquire data	$(I_{UC_1} + I_{UC_{ACQ}} + I_{RM_0}) \times t_{UC_{ACQ}}$	$(I_{SoC_2} + I_{SoC_{ACQ}}) \times t_{SoC_{ACQ}}$
	Change power mode	$Q_{UC_{1\to0}}$	$Q_{SoC_{2\to0}}$
Report data	Change power mode	$Q_{UC_{0\to2}} + Q_{RM_{0\to2}}$	$Q_{SoC_{0\to2}}$
	Exchange "report_data" command (RM → UC)	$(I_{UC_2} + I_{UC_{SCI}} + I_{RM_2} + I_{RM_{SCI}}) \times t_{SCI_{REPORT}}$	0
	Change power mode	$Q_{UC_{2\to0}} + Q_{RM_{2\to3}}$	$Q_{SoC_{2\to3}}$
	Send data to the network (rejoin if needed)	$(I_{UC_0} \times t_{REPORT}) + Q_{RM_{REJOIN}} + Q_{RM_{REPORT}}$	$Q_{SoC_{REJOIN}} + Q_{SoC_{REPORT}}$
	Change power mode	$Q_{RM_{3\to0}}$	$Q_{SoC_{3\to0}}$
Poll parent	Change power mode	$Q_{UC_{0\to2}} + Q_{RM_{0\to2}}$	$Q_{SoC_{0\to2}}$
	Exchange Poll event (UC → RM)	$(I_{UC_2} + I_{UC_{SCI}} + I_{RM_2} + I_{RM_{SCI}}) \times t_{SCI_{POLL}}$	0
	Change power mode	$Q_{UC_{2\to1}} + Q_{RM_{2\to3}}$	$Q_{SoC_{2\to3}}$
	Poll parent in the network	$(I_{UC_1} \times t_{POLL}) + Q_{RM_{POLL}}$	$Q_{SoC_{POLL}}$
	Change power mode	$Q_{UC_{1\to2}} + Q_{RM_{3\to2}}$	$Q_{SoC_{3\to0}}$
	Exchange "poll response" (RM → UC)	$(I_{UC_2} + I_{UC_{SCI}} + I_{RM_2} + I_{RM_{SCI}}) \times t_{SCI_{POLL\_ANSW}}$	0
	Change power mode	$Q_{UC_{2\to0}} + Q_{RM_{2\to0}}$	
Sleep	Sleep	$(I_{UC_0} + I_{RM_0}) \times t_{SLEEP}$	$I_{SoC_0} \times t_{SLEEP}$

Table 4: Figures involved in the calculation of power consumption.

	Dual architecture				Single architecture	
	uC_{PIC}	RM_{Ember}	uC_{TI}	RM_{TI}	SoC_{Ember}	SoC_{TI}
I_0	0.835 μA	0.4 μA	0.9 μA	0.4 μA	1 μA	1 μA
I_1	15 μA	—	41 μA	—	—	—
I_2	3.05 mA	6 mA	2.2 mA	3.4 mA	6 mA	3.4 mA
I_3	—	27 mA	—	28.7 mA	27 mA	28.7 mA
I_{ACQ}	1 mA	—	850 μA	—	1.1 mA	1.2 mA
I_{SCI}	0.5 μA	200 μA	0.5 μA	200 μA	—	—
$Q_{0\to1}$	15 pC	—	16 pC	—	—	—
$Q_{0\to2}$	0.39 μC	12.4 μC	10 pC	51.57 μC	12.4 μC	51.57 μC
$Q_{2\to3}$	—	9.94 μC	—	40.95 μC	9.94 μC	40.95 μC
$Q_{3\to0}$	—	3.3 μC	—	13.6 μC	3.3 μC	13.6 μC
$t_{0\to1}$	1 μs	—	0.4 μs	—	—	—
$t_{0\to2}$	128 μs	—	0.4 μs	—	—	—
t_{ACQ}	4.125 μs	—	2.06 μs	—	42.7 μs	68 μs
t_{SCI_POLL}	16 μs	16 μs	8 μs	8 μs	—	—
$t_{SCI\_POLL\_ANSW}$	4 μs	4 μs	2 μs	4 μs	—	—
t_{SCI_REPORT}	34 μs	34 μs	17 μs	34 μs	—	—
t_{REPORT}	—	8 ms	—	8 ms	8 ms	8 ms
t_{POLL}	—	6 ms	—	6.8 ms	6 ms	6.8 ms

(iii) Current in power mode 0 of uC, RM, and SoC is several orders of magnitude lower compared to power modes 1, 2, or 3.

(iv) Time in sleep mode is several orders of magnitude larger than any other times.

Considering the former simplifications and application times between events (t_{SAMPLE}, t_{REPORT}, and t_{POLL}), the resulting energy balance between dual and single architecture for a given cycle is

$$Q_{CYCLE_{D-S}} = \frac{t_{REPORT}}{t_{SAMPLE}} \times Q_{SENSE_{D-S}} + \frac{t_{REPORT}}{t_{POLL}} \times Q_{POLL_{D-S}} + Q_{REPORT_{D-S}} + t_{REPORT} \times I_{SLEEP_{D-S}}, \quad (2)$$

where

$$\begin{aligned} Q_{SENSE_{D-S}} &= Q_{uC_{0\to1}} + Q_{uC_{1\to0}} + I_{RM_0} \\ &\quad \times \left(t_{uC_{0\to1}} + t_{uC_{ACQ}}\right) + \left(I_{uC_1} + I_{uC_{ACQ}}\right) \\ &\quad \times t_{uC_{ACQ}} - Q_{SoC_{0\to2}} - Q_{SoC_{2\to0}} \\ &\quad - \left(I_{SoC_2} + I_{SoC_{ACQ}}\right) \times t_{SoC_{ACQ}} \\ &\quad + \left(I_{RM_0} + I_{UC_1} - I_{SoC_2}\right) \times t_{SENSOR}, \end{aligned}$$

$$\begin{aligned} Q_{POLL_{D-S}} &= Q_{uC_{0\to2}} + Q_{uC_{2\to0}} + Q_{uC_{1\to2}} + Q_{uC_{2\to1}} \\ &\quad + \left(I_{uC_2} + I_{uC_{SCI}} + I_{RM_2} + I_{RM_{SCI}}\right) \\ &\quad \times \left(t_{SCI_{POLL}} + t_{SCI_{POLL_{ANSW}}}\right) + \left(I_{UC_1} \times t_{POLL}\right), \\ Q_{REPORT_{D-S}} &= Q_{UC_{0\to2}} + Q_{UC_{2\to0}} \\ &\quad + \left(I_{UC_2} + I_{UC_{SCI}} + I_{RM_2} + I_{RM_{SCI}}\right) \\ &\quad \times t_{SCI_{REPORT}} + \left(I_{UC_0} \times t_{REPORT}\right), \\ I_{SLEEP_{D-S}} &= I_{UC_0} + I_{RM_0} - I_{SoC_0}. \end{aligned} \quad (3)$$

Thus, when $Q_{CYCLE_{D-S}} < 0$, the dual architecture will be more power efficient than the single architecture and vice versa when $Q_{CYCLE_{D-S}} > 0$.

4. Experimental Method and Results

As mentioned above, there are different WSN simulation tools that focus on specific aspects of the network: latency times, bandwidth, collisions, message integrity, and so forth. According to the previous section analysis, we need to focus more deeply on the architecture of the node and associated states, than on the network characteristics. Thus, we used MATLAB suite to model energy consumption of real sensing nodes' hardware and simulate FSM operation.

Comparison between architectures has been done analyzing two real implementations with devices having similar

Table 5: Rate of consumption of each substate (considering that $t_{\text{report}} = 8$ hours, $t_{\text{SAMPLE}} = 10$ min, $t_{\text{POLL}} = 4$ min, $t_{\text{SENSOR}} = 10$ ms).

	Microchip-Ember	Texas Instruments
$\%Q_{\text{SENSE}_{D-S}}$	33.707%	32.094%
$\%Q_{\text{POLL}_{D-S}}$	0.773%	0.315%
$\%Q_{\text{REPORT}_{D-S}}$	0.007%	0.002%
$\%(I_{\text{SLEEP}_{D-S}} \times t_{\text{REPORT}})$	65.513%	67.589%

characteristics: both ZigBee standard chipsets and microcontrollers with 16 bit RISC architecture similar to MIPS, power supply ranges, integration of peripherals (ADC, serial communication interfaces, clocks, etc.), and memory capacity. Table 4 shows how theoretical analysis shown in Section 3 is specified for two different implementations of ZigBee standard (Texas Instruments and Ember, but now Silicon Labs) and for two different families of ultralow power microcontrollers (Microchip and Texas Instruments). (uC_{PIC} = PIC24F16KA102; $RM_{\text{Ember}} = SoC_{\text{Ember}}$ = EM357; uC_{TI} = MSP430F2001; $RM_{\text{TI}} = SoC_{\text{TI}}$ = CC2530. SoC manufacturers usually allow their devices to operate as RM running a specific firmware. Thus, in order to eliminate hardware dependencies in analysis, we decided to use the same chipset operating in different configurations in both architectures. The indicated energy consumption corresponds to the scenarios in which both architectures have optimized and similar performance: similar peripheral, clocks sources, and power configuration. It is important to remark that internal RTCC in PM_0 has been selected.)

For a given conditions and according to the analysis in Section 3, Table 5 shows the charge difference between dual and single architecture ($\%Q_{X_{D-S}}$) of each substate, expressed in percentage contribution to the normalized total consumption per cycle. On one hand, it highlights the importance of sleeping and sensing processes related to total energy consumption evidencing their importance in autonomy maximization. It also proves the slight differences between chipsets, which together with the fact that information available about power consumption is more profuse for Microchip-Ember configuration leads us to choose it for further analyses.

4.1. Sensing and Reporting. When focusing on measurement process, there are two important tasks: data acquisition and reporting. Figure 3 represents how the power savings ratio (PSR) of the dual architecture versus single architecture (defined as $\text{PSR}_{\text{DSvsS}} = Q_{\text{CYCLE}_{D-S}}/Q_{\text{CYCLE}_S} \triangleq \Delta Q/Q$) varies depending on t_{SAMPLE}, t_{POLL}, and t_{SENSOR}. Values above zero indicate better performance of the dual architecture and vice versa when $\text{PSR}_{\text{DSvsS}}$ is below zero.

It is appreciated that variation in t_{POLL} has reduced impact on $\text{PSR}_{\text{DSvsS}}$. The major effect comes from the variation of the time between measurements (t_{SAMPLE}) and the time needed to have valid sensor signal (t_{SENSOR}) [34]; the more time the node spends in sensing tasks, the more effective the dual architecture becomes. This fact is evidenced

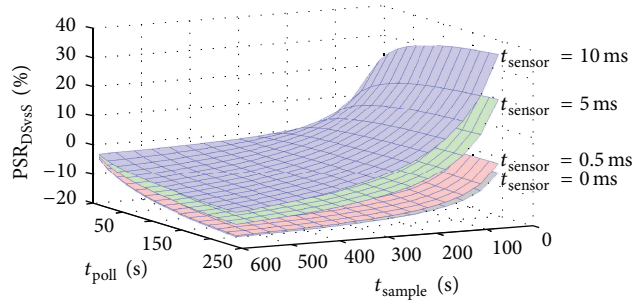

Figure 3: Variation of PSR_{DvsS} with t_{SAMPLE} and t_{POLL} (e.g., $t_{\text{REPORT}} = 4$ hours and several t_{sensor}).

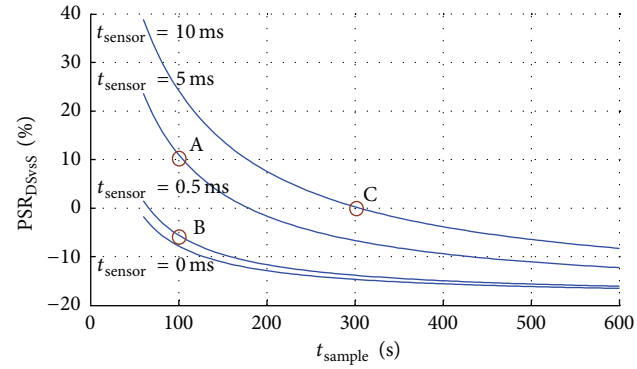

Figure 4: $\text{PSR}_{\text{DSvsS}}$ versus t_{SAMPLE} (e.g., $t_{\text{REPORT}} = 4$ hours $t_{\text{POLL}} = 4$ min).

in Figure 4, where $\text{PSR}_{\text{DSvsS}}$ is represented versus t_{SAMPLE} for various values of t_{SENSOR}.

We can clearly observe the impact of the measurement process on energy savings in the following example. Considering a sensor node getting one sample each 100 seconds from a sensor that needs 5 ms to provide a valid value (point A in Figure 3), the dual architecture would need 10% of energy less than single architecture. This effect is mainly derived from the higher flexibility in terms of clock sources of low power microcontrollers that is so far not available in SoCs (PIC24F16KA102 has five external and internal clock sources, providing 11 different clock modes with a minimum CPU clock speed of 31 kHz. Ember 357 has four clock sources with a minimum CPU clock speed of 6 MHz. The same happens to TI's hardware); that is, microcontrollers consider low power modes with slow clocks (PM_1) that are very convenient for sensing tasks. On the other hand if t_{SENSOR} is reduced to 500 us (point B in Figure 3), single architecture would be 6% more efficient. Finally, when sampling time t_{SAMPLE} exceeds 5 minutes (point C in Figure 3), for the conditions given ($t_{\text{REPORT}} = 4$ hours; $t_{\text{POLL}} = 4$ min; $t_{\text{SENSOR}} \leq 10$ ms), single architecture will be always more efficient.

4.2. Rejoining and Polling Strategies. Regardless of the dual or single architectures, if it is assumable that the node is not connected to the network, a rejoin strategy can be more

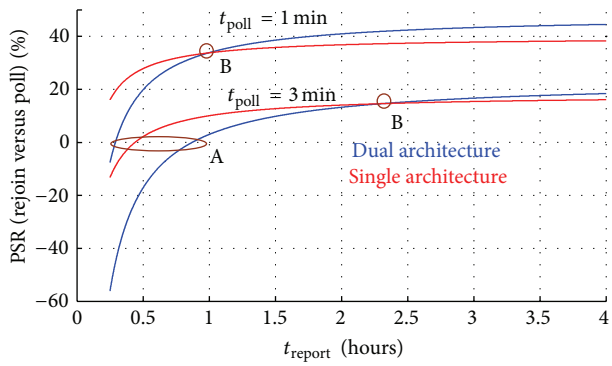

Figure 5: Comparison and single dual architecture with rejoining versus polling strategies (example for t_{SAMPLE} = 1 min t_{SENSOR} = 5 ms).

optimal depending mainly on the reporting period (t_{REPORT}). This basically occurs when the overconsumption due to rejoin process compensates the accumulated energy consumptions of the polls. Figure 5 compares PSR between rejoining and polling strategies for single and dual architectures.

Intersection between lines with zero (points A in Figure 4) indicates the t_{REPORT} above which rejoining strategy would be more convenient for any architecture. Intersection between red and blue lines (points B in Figure 4) indicates the t_{REPORT} above which dual architecture is more efficient than single architecture.

As expected, the energy savings of rejoining strategy increases with time between reports, faster at the beginning, until reaching a final stable value. This is because increasing time between reports decreases relative impact of Q_{REJOIN} over the total. For this same reason, the final PSR is much more affected by the time between polls rather than by the value of Q_{REJOIN}.

4.3. Sleeping. As we have seen in Table 5, with any given sampling/polling/reporting conditions, the current in sleep mode is a relevant variable that has major impact in node lifetime. Thus, it is evident that the primary goal of a low power system is being in sleep mode as long as possible [35]. Some authors propose adaptive runtime to maximize efficiency [36]. Indeed, it is common to perform nodal power consumption analysis according to sleeping duty cycle [37]. Given the presented FSM tasks, considering sleeping time that is several orders of magnitude higher than the time devoted to all other tasks, having a battery charged with Q_{BATT} and "n" being the number of reports performed by the node during its lifetime, charge will be drained as

$$Q_{BATT} = Q_{INIT} + n \times \left(\frac{t_{REPORT}}{t_{SAMPLE}} \times Q_{SENSE} + \frac{t_{REPORT}}{t_{POLL}} \times Q_{POLL} + Q_{REPORT} + t_{REPORT} \times I_{SLEEP} \right). \quad (4)$$

Dual architecture with low power microcontrollers allows greater versatility to reduce sleep current, due to additional capabilities provided by a microcontroller: ultralow wakeup with external capacitor and radio module's totally powered off. (Frequently, microcontrollers have external interrupts based on discharged time of a capacitor. (See Microchip AN879 Using the Microchip Ultra Low-power Wake-up Module) or high impedance RC external circuits could be used in an low power interrupt. Note that the consumption for charging this capacitor is negligible.) Both architectures can also use an external RTCC to reduce to the maximum energy required for timing. (Low-Current High-ESR Crystals (such as Maxim DS1341) with I^2C communication and one output used to activate an alarm interrupt of the microcontroller.) Table 6 shows pros and cons of different sleep mode strategies, sleep current of hardware, and associated PSR of dual architecture versus single architecture.

For polling (node can receive messages) and rejoining (node cannot receive messages) configurations, we considered four sleeping strategies. Using internal or external RTCC (additional chip necessary) provides node's conscience about clock and calendar and high precision in wakeup timing. It can be useful to build time synchronized WSNs, to accurately monitor variables or to timestamp measurements. Internal WDT reduces current consumption and loses timing functionalities. Finally, ultralow power wakeup has the most inaccurate timing (that could be enough to form any applications) but greatly reduces current consumption.

Evidently, the more the silicon modules that can be powered off, the less the power consumption in sleep mode. Thus, due to its higher flexibility, the dual architecture can be very convenient in case the application requirements allow it; it is especially remarkable to note the PSR difference in the rejoining strategy with ultralow power wakeup.

4.4. Hardware Architecture Performance Comparison. In order to range the importance of the issues described here, this section provides a hardware architecture performance comparison of well-known WSN platforms [38–40]. The methodology followed has been to model the hardware blocks of the platforms according to chip manufacturer specifications and calculate the expected battery lifetime in a realistic scenario. Table 7 show the life expectancy expressed in years and the ratio compared to the best performance architecture. (Test framework considered: V_{supply} = 3 V; internal oscillator, main frequency = 8 mhz, secondary frequency = 1 MHz; External Oscillator, Crystal frequency = 32.768 kHz; t_{SAMPLE} = 120 s, t_{POLL} = 4 min, t_{SENSOR} = 1 ms; t_{REPORT} = 60 min; Battery type = $LiMnO_2$, model = 2032/5004LC, capacity = 210 mAh). Obviously, it is necessary to consider that older systems are at disadvantage as chipset performance improves every year.

According to the results in previous sections, dual architecture is more efficient than the single one for the given conditions. Also both Texas Instruments and Microchip-Ember provides the highest performance. As sensing duties are not exigent in terms of microcontroller requirements, we can observe the negative effect of oversizing them (SunSpot's microcontroller is very powerful) in terms of life expectancy.

Table 6: PSR_{DSvsD} for several configurations (t_{SAMPLE} = 120 s, t_{POLL} = 4 min, and t_{SENSOR} = 1 ms).

	Sleeping strategy	I_{SLEEP} (Dual)	I_{SLEEP} (Single)		PSR_{cycle} (DSvsS)	Pros and cons
		uC_{PIC} + $RTCC_{DS}$	RM_{Ember}	SoC_{Ember}		
Polling	Internal RTCC wakeup	0.835 μA	0.4 μA	1 μA	5.68%	+ Node can receive messages + High precision in wakeup timing
	Internal WDT wakeup	0.585 μA	0.4 μA	0.8 μA	2.58%	+ Node can receive messages – Low precision in wakeup timing
	External RTCC wakeup	0.035 μA + 0.25 μA	0.4 μA	0.4 μA + 0.25 μA	0.78%	+ Node can receive messages + High precision in wakeup timing – Additional RTCC chip necessary
	Ultralow power wakeup	0.035 μA	0.4 μA	0.8 μA	–41.63%	+ Node can receive messages – The lowest precision in wakeup timing
Rejoining	Internal RTCC wakeup	0.835 μA	0	1 μA	–25.90%	– Node cannot receive messages + High precision in wakeup timing
	Internal WDT wakeup	0.585 μA	0	0.8 μA	–35.94%	– Node cannot receive messages – Low precision in wakeup timing
	External RTCC wakeup	0.035 μA + 0.25 μA	0	0.4 μA + 0.25 μA	–59.47%	– Node cannot receive messages + High precision in wakeup timing – Additional RTCC chip necessary
	Ultra low power wakeup	0.035 μA	0	0.8 μA	–90.66%	– Node cannot receive messages – The lowest precision in wakeup timing

Table 7: WSN hardware platform performance comparison.

	Platform	Hardware architecture		Life expectancy (Years)	Ratio (%)
		Microcontroller	Transceiver		
Dual	Texas Instruments	MSP43F2001	CC2530	2.75	100%
	Microchip-Ember	PIC24F16KA102	EM357	2.45	89.12%
	Iris-It (2008)	ATMega 1281	AT86RF230	1.03	37.42%
	Libelium (2012)	ATMega 1281	EM357	0.99	36.16%
	TelosB, Shimmer (2005)	MSP430F1611	CC2420	0.75	27.47%
	MicaZ (2004)	Atmega 128L	CC2420	0.42	15.11%
	Sun SPOT (2007)	AT91SAM9G20	CC2420	0.14	5.20%
Single	Texas Instruments	CC2530		2.13	74.34%
	Ember	EM357		1.42	48.90%

Also, comparing performance of platforms sharing the same transceiver (CC2420 and EM357), the influence of the microcontroller chosen is obvious.

5. Conclusions

WSNs are essential in the next generation of Internet where ubiquitous interconnected objects are available for interaction. Ubiquity means everywhere and anytime availability of sensing nodes implying wireless communication, energy harvesting, low power, and so forth; concepts that if not properly considered can lead to reduced systems' autonomy killing many real IoT applications. With these considerations in mind, low power consumption is one of the most important targets when designing IoT ready sensors.

This paper studies different sensor node hardware architectures, deepening in the power consumptions associated with each state of the runtime cycle and time-relationship between them. It compares the energy consumption involved in the operation of a sensor node implemented using two different architectures: dual (based on a low power microcontroller and a radio module) and single (based on a system on chip). The specific finite state machine that describes the operation of sensing node is based on standard IEEE 1451 and the specific communications substates are modeled according to ZigBee Pro standard.

One important conclusion is that energy required in the sensing procedure has an important impact on this balance. There are some tasks, such as waiting for a valid sensor output (t_{SENSOR}) or acquiring the sensor data, which might require relevant amount of energy depending on the sampling rate (t_{SAMPLE}). This can turn dual architecture more efficient than the single one. One reason is that because low power microcontrollers in single architecture have higher flexibility than SoC architectures in terms of low power oscillator configurations, microcontrollers embedded in SoCs are usually

not able to run with kHz oscillators. The second reason is because low power microcontroller peripherals are more optimized, something which can be especially relevant in case of using analog sensors that require the use of analog-digital converter (the same performance in terms of quality of the conversion requires less current and time in low power microcontroller than in SoC).

Considering temporal ubiquity requirements, if the IoT application does not require nodal availability at any time (for example to change sampling parameters), nodes can disconnect from the WSN. In case of ZigBee standard, this can be implemented using rejoining and polling strategies. In that case, when energy needed to rejoin exceeds consumption due to several polls, polling strategy turns to be more energy efficient. It also shows that, above a certain reporting period, dual architecture is more efficient because rejoining strategy allows to totally power off the radio module when not using it.

Power consumption in sleep mode has major impact on node lifetime, so there is a need to design a system with a current in sleep mode as low as possible. Again, dual architecture might be more convenient because low power microcontrollers are more flexible in terms of oscillator configuration and have additional low power modules such as ultralow-power wake-up module.

The main conclusion of the study evidences that, despite what could be considered initially and stated in datasheets, no architecture is always energetically more efficient than the other; deep contextualized system analysis is mandatory to squeeze batteries to the maximum. This paper provides generic guidelines that would help electronic designers in this analysis in order to decide the most energy efficient hardware architecture of sensor nodes. We also find it useful for firmware and even software developers in order to provide understanding about how IoT application requirements (e.g., reporting time) affect WSN performance and lifetime. Finally, a performance comparison of different WSN platforms attending to their hardware architecture evidences the impact of the issues just stated.

As a final example, making clear the importance of the analysis, if a sensor that polls for data every 4 min samples every minute a sensor that needs 5 ms to set up and reports data each 4 hours is implemented using a dual architecture, it would need 24% less energy than implemented using a SoC. But just changing sampling rate from 1 minute to 5 minutes would turn the situation making the dual architecture consume 6% more energy than single architecture.

References

[1] O. Mirabella and M. Brischetto, "A hybrid wired/wireless networking infrastructure for greenhouse management," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 2, pp. 398–407, 2011.

[2] O. A. Postolache, P. M. B. S. Girao, J. Mendes, E. C. Pinheiro, and G. Postolache, "Physiological parameters measurement based on wheelchair embedded sensors and advanced signal processing," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 10, pp. 2564–2574, 2010.

[3] V. C. Gungor and G. P. Hancke, "Industrial wireless sensor networks: challenges, design principles, and technical approaches," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 10, pp. 4258–4265, 2009.

[4] C. C. Gómez, J. Paradells, and J. E. Caballero, *Sensors Everywhere. Wireless Network Technologies and Solutions*, Fundación Vodafone España, 2010.

[5] A. Sharma, K. Shinghal, R. Singh, and N. Srivastaya, "Energy management for wireless sensor network nodes," *International Journal of Advances in Engineering & Technology*, vol. 1, pp. 7–13, 2011.

[6] C. Knight, J. Davidson, and S. Behrens, "Energy options for wireless sensor nodes," *Sensors*, vol. 8, no. 12, pp. 8037–8066, 2008.

[7] R. Soua and P. Mine, "A survey on energy efficient techniques in wireless sensor networks," in *Proceedings of the Wireless and Mobile Networking Conference (WMNC '11)*, INRIA, Le Chesnay, France, 2011.

[8] J. E. Higuera and J. Polo, "IEEE 1451 standard in 6LoWPAN sensor networks using a compact physical-layer transducer electronic datasheet," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 8, pp. 2751–2758, 2011.

[9] IEEE Standard 1451.5, "IEEE standard for a smart transducer interface for densors and actuators—wireless communication protocols and transducer electronic data sheet (TEDS) formats," IEEE, 2007.

[10] C. Buratti, A. Conti, D. Dardari, and R. Verdone, "An overview on wireless sensor networks technology and evolution," *Sensors*, vol. 9, no. 9, pp. 6869–6896, 2009.

[11] ZigBee Alliance, "ZigBee Specification," Document 053474r17, 2008.

[12] R. X. Gao and Z. Fan, "Architectural design of a sensory node controller for optimized energy utilization in sensor networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 55, no. 2, pp. 415–428, 2006.

[13] S. Bilouhan and R. Gupta, "Optimization of power consumption in wireless sensor networks," *International Journal of Scientific & Engineering Research*, vol. 2, no. 5, 2011.

[14] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Networks*, vol. 3, no. 3, pp. 325–349, 2005.

[15] M. Al Ameen, S. M. R. Islam, and K. Kwak, "Energy saving mechanisms for MAC protocols in wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2010, Article ID 163413, 16 pages, 2010.

[16] S. M. Ahmad Madani, *Cross layer design for low power wireless sensor networks [Ph.D. thesis]*, 2008.

[17] P. Yu, L. Qinghua, and P. Xiyuan, "The design of low-power wireless sensor node," in *Proceedings of the IEEE International Instrumentation and Measurement Technology Conference (I2MTC '10)*, pp. 917–922, May 2010.

[18] T. R. Park and M. J. Lee, "Power saving algorithms for wireless sensor networks on IEEE 802.15.4," *IEEE Communications Magazine*, vol. 46, no. 6, pp. 148–155, 2008.

[19] F. Salvadori, M. de Campos, P. S. Sausen et al., "Monitoring in industrial systems using wireless sensor network with dynamic power management," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 9, pp. 3104–3111, 2009.

[20] S. Caban, G. José Antonio, and M. Rupp, "Measuring the physical layer performance of wireless communication systems," *IEEE Instrumentation & Measurement Magazine*, vol. 14, no. 5, pp. 8–17, 2011.

[21] C. Alippi, G. Anastasi, M. Di Francesco, and M. Roveri, "Energy management in wireless sensor networks with energy-hungry sensors," *IEEE Instrumentation and Measurement Magazine*, vol. 12, no. 2, pp. 16–23, 2009.

[22] Texas Instruments Incorporated, http://www.ti.com/product/cc2420.

[23] "XP Semiconductors N.V.," http://www.nxp.com/campaigns/i2c-bus/.

[24] "Silicon Laboratories Inc.," http://www.silabs.com/products/wireless/zigbee/Pages/zigbee-chips-em35x.aspx.

[25] "Texas Instruments Incorporated," http://www.ti.com/product/cc2530.

[26] P. H. Chou and C. Park, "Energy-efficient platform designs for real-world wireless sensing applications," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD '05)*, pp. 913–920, November 2005.

[27] M. Gao, X. Pan, L. Deng, C. Huang, D. Zhang, and L. Ni, "A versatile nodal energy consumption monitoring method for wireless sensor networks testbed," in *Proceedings of the IEEE 17th International Conference on Parallel and Distributed Systems*, pp. 388–395, 2011.

[28] Castalia, http://castalia.research.nicta.com.au/index.php/en/.

[29] G. V. Merrett, N. M. White, N. R. Harris, and B. M. Al-Hashimi, "Energy-aware simulation for wireless sensor networks," in *Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON '09)*, June 2009.

[30] P. Yu, L. Qinghua, and P. Xiyuan, "The design of low-power wireless sensor node," in *Proceedings of the IEEE International Instrumentation and Measurement Technology Conference (I2MTC '10)*, pp. 917–922, May 2010.

[31] H.-Y. Zhou, D.-Y. Luo, Y. Gao, and D.-C. Zuo, "Modeling of node energy consumption for wireless sensor networks," *Wireless Sensor Networks*, vol. 3, no. 1, pp. 18–23, 2011.

[32] E. Casilari, J. M. Cano-García, and G. Campos-Garrido, "Modeling of current consumption in 802.15.4/ZigBee sensor motes," *Sensors*, vol. 10, no. 6, pp. 5443–5468, 2010.

[33] B. Gholamzadeh and H. Nabovati, "Concepts for designing low power wireless sensor network," *World Academy of Science, Engineering and Technology*, vol. 45, pp. 559–565, 2008.

[34] W. S. Wang, R. O'Keeffe, N. Wang, M. Hayes, B. O. 'Flynn, and S. C. Ó Mathúna, "Reducing power consumption in metrics for building energy management apllications," in *Proceedings of the 23rd European Conference Forum Bauinformatik*, Cork, Ireland, 2011.

[35] A. Viswanathan and T. E. Boult, "Power conservation in ZigBee networks using temporal control," in *Proceedings of the 2nd International Symposium on Wireless Pervasive Computing*, pp. 175–180, University of Colorado Press, Boulder, Colo, USA, February 2007.

[36] J. Chern Lim and C. Bleakley, "AdaptiveWSN scheduling for lifetime extension in environmentalMonitoring applications," *International Journal of Distributed Sensor Networks*, vol. 2012, Article ID 286981, 17 pages, 2012.

[37] Y. W. Chung and H. Y. Hwang, "Modeling and analysis of energy conservation scheme based on duty cycling in wireless Ad Hoc sensor network," *Sensors*, vol. 10, no. 6, pp. 5569–5589, 2010.

[38] V. Madan and S. Reddy, "Review of wireless sensor mote platforms," *VSRD International Journal of Elecrical, Electronics & Communication Engineering*, vol. 2, no. 2, pp. 50–55, 2012.

[39] T. V. Chien, H. Nguyen Chan, and T. Nguyen Huu, "A comparative study on hardware platforms for wireless sensor networks," *International Journal on Advanded Science Engineering Information Technologgy*, vol. 2, no. 1, 2012.

[40] R. Lajara, J. Pelegrí-Sebastiá, and J. J. Perez Solano, "Power consumption analysis of operating systems for wireless sensor networks," *Sensors*, vol. 10, no. 6, pp. 5809–5826, 2010.