A Simulation Model for Wireless Sensor Networks Based on TOSSIM

Francisco J. Rincón[1], Alexandru E. Şuşu[2], Marcos Sánchez-Élez[1], David Atienza[1,2], Giovanni de Micheli[2]

[1] DACYA, Complutense University of Madrid (UCM), Madrid, Spain
francisco.rincon@fdi.ucm.es, marcos@fis.ucm.es, datienza@dacya.ucm.es
[2] LSI, EPFL, Lausanne, Switzerland, {alex.susu, david.atienza, giovanni.demicheli}@epfl.ch

Abstract
Wireless Sensor Networks (WSNs) are composed of a large number of very small devices used for biomedical or environmental monitoring applications. These devices measure various natural processes at different locations, and cooperate in order to aggregate the information. In this paper we propose an accurate model based on the *TinyOS simulator (TOSSIM)* for wireless sensor devices, which can be used to precisely estimate the network performance or the power consumption of real-life sensor nodes. This model helps us to evaluate the impact on network performance of making changes in the node architecture or in the communication layers. To show the efficiency and versatility of our model we have tested it with the Sensor Cube nodes, developed by IMEC, and measured and tuned the power consumption for different *Medium Access Control (MAC)* protocols.

Keywords
Wireless Sensor Network, MAC protocol, Operating System.

1. Introduction
Wireless Sensor Networks [1] (WSN) are networks composed of spatially distributed autonomous nodes equipped with radio transmitters, which cooperatively monitor physic magnitudes. Their utility ranges from environmental measurements at different locations to medical monitoring. The network's main tasks are to sense the relevant information at every node, to collect that information, to process and communicate it to a base station. In order to perform these activities each node is equipped with a sensor, a radio transceiver and a microcontroller. As power source they usually have a battery, but energy harvesting modules (such as solar cells or thermoelectric converters) can also be included, increasing the life of the node and therefore minimizing its maintenance [11].

Many types of sensors can be integrated in a node, and, therefore, the number of applications for WSNs is very large. For instance, temperature, humidity or illumination sensors can be used for environmental monitoring [3], *Electrocardigram/electroencephalogram (ECG/EEG)* sensors for healthcare applications [13], and activity sensors for home automation. We can also use these networks for traffic control, seismic or fire detection, sports monitoring, etc. The WSNs are used in commercial and industrial applications to monitor data that would be difficult or expensive to monitor using wired sensors. They can be deployed in wilderness areas, where they would remain for many years (monitoring some environmental variables) without the need to recharge/replace their power supplies.

Each node needs computational power in order to process the sensed data and perform the routing functions in the network. It is desirable to have a decentralized network. Using this distributed processing, the communication between nodes, and therefore the power consumption, is significantly reduced, partly because the microcontroller and memory elements consume severely less energy than the radio transmitter.

Using a radio chip instead of a wired network interface gives the desired mobility to the nodes. Manufacturers develop new micro-radios with lower power consumption and adapt them to the needs of sensor networks [14]. Some of these transceivers implement MAC protocols such as Bluetooth and ZigBee in hardware.

Many parameters can be changed to make the network more suitable to the application. For example, we can adapt the network topology to improve one particular monitoring characteristic; or we can use different *Medium Access Control (MAC)* [2] and routing protocols with the goal of reducing the power consumption. We can also consider the possibility to transmit internal information of the node to its neighbors, for instance, to inform them about the low battery charge. Also the application can be dynamically adjusted. If we have a node equipped with temperature and humidity sensors, we could envision, for example, that when the humidity is peaking, we should sense more frequently the temperature."

In order to measure the impact of changes in the configurable parameters of the network on the performance and estimate power consumption, reliable simulators and therefore reliable models of sensors architectures are highly needed. We can make changes in the WSN architecture model for simulation

before making them in the real platform. For example, if we want to test a new MAC protocol we can implement it, debug it and test it in the simulator to get values of its impact in both performance and power consumption, before implementing it in the real node.

The paper begins with a description of the Sensor Cubes, the platform used to validate our model, describing in more detail the radio transceiver in section 3, since this is the most power hungry component of the platform. In section 4 we present a brief overview of the operating system that is used in the nodes and the simulator. Section 5 presents the power model that we propose, which is analyzed with a case study in section 6. Finally, conclusions are discussed in section 7.

2. Sensor Cubes

In order to validate our model we use the Sensor Cube platform, developed by IMEC-NL/Holst [3], a 14x14x18mm^3 sensor node. We present the platform in figure 1, placed on a 2 Euro coin.

The Sensor Cube is divided in different layers with different functionalities. The top layer is responsible for the wireless communication. It consists of a single chip short-range 2.4GHz transceiver (Nordic nRF2401, 18 nJ/bit) with a coplanar integrated folded dipole antenna [8].

The next layer contains an MSP430 microcontroller, which controls the entire module [7]. The microcontroller has a very low active power (0.6 nJ/instruction), low standby power (2μW), fast wakeup from standby to active mode (6μs) and on-chip 12-bit analog-to-digital converter. A 32.768 KHz ±20ppm crystal provides the system's local time reference.

A power management layer has been implemented, which accepts power from an energy scavenger such as a compact solar cell or thermoelectric generator, and uses it for keeping a secondary battery charged.

The final component of the sensor node is the sensor layer which measures different parameters, depending on the kind of sensors: temperature, humidity or illumination for environmental monitoring or electrocardiogram, electroencephalogram, etc. for medical applications.

IMEC-NL/Holst has also developed a USB stick-like base station, shown in figure 2. It is connected to a PC and receives the data from all the sensor nodes in the network. This USB stick has the same architecture as the nodes, except that it does not include the sensors and the power management sub-systems, since it is meant to collect the data from the network and, respectively, it consumes energy from the PC it is connected to.

Figure 1. Sensor Cube on a 2 Euro coin

Figure 2. USB stick

3. Wireless Communications: Nordic nRF2401

In order to develop an accurate simulation tool we need to study all the layers of the node. However, since the component of the node that consumes more power is the radio transceiver, we will focus our modeling efforts onto it.

The radio chip inserted in the sensor cube has two modes of operation: direct and ShockBurst, which are described below.

In *direct mode* the radio works like a traditional RF device. The supported data rates are 1Mbps ±200ppm or 250Kbps ±200ppm.

The *ShockBusrt* technology uses the on-chip FIFO to clock in data at a low data rate and transmit at a very high rate, which enables an aggressive energy consumption reduction. We can use a low frequency crystal to clock in data, while the transmission rate is very high. Another advantage of this mode is the reduction of the processing in the microcontroller, because the radio chip also checks the address and *CRC (Cyclic Redundancy Check)* of the packets received. In this operation mode, before sending a packet, the radio adds to it the following information: the preamble, which allows the receiver to detect the incoming data (and recover the clock if we are using direct mode of operation), the destination address and the CRC. The receiving node checks the information of the ShockBurst data frame, and if the destination address of the packet is not the same as the node address, the packet is discarded; in the same way, the packet is discarded if the CRC check is not correct.

The structure of the ShockBurst packet is depicted in figure 3. The preamble is 8 bits long, the address can be from 8 to 40 bits long, the payload size is 256 bits (at maximum) from which we subtract the following: the address, which can have from 8 to 40 bits, and the CRC, which is optional, and it can have 8 or 16 bits. In our case we use 8 bits for the *preamble*, 24 bits for the *address*, 216 bits for the *payload* and 16 bits for the CRC.

PREAMBLE	ADDRESS	PAYLOAD	CRC

Figure 3. Data packet diagram

In our model we only consider the ShockBurst mode because it is the only mode included in the real platform used as case study, since there is no need for communication from the base station to the nodes. Thus, this implementation using the low frequency crystal included in the design achieves the best low power consumption.

4. TOSSIM

TinyOS [6] is an open-source, event-based operating system designed for wireless sensor networks, which is extensively

supported in the research and industrial communities. When an external event occurs, such as an incoming data packet or a sensor reading, TinyOS calls the appropriate event handler to process the newly available data.

This operating system is written using the NesC programming language [12], which is an extension to the C language optimized for the memory restrictions of sensor networks. TinyOS is designed to support the concurrent-intensive operations required by networked sensors with minimal hardware requirements.

TOSSIM [4] is a discrete event simulator for TinyOS sensor networks and is part of our power simulation tool. Instead of compiling a TinyOS application for the nodes, the user can compile it using the TOSSIM framework, and then run the application on a PC. This allows users to debug, test, and analyze the implemented code in a controlled and repeatable environment. TOSSIM can simulate thousands of nodes simultaneously. Every mote in a simulation runs the same TinyOS program.

The latest version of the simulator is called PowerTOSSIM. The main contribution w.r.t. the previous versions of TOSSIM is the inclusion of the assessment of the power consumed by the application. PowerTOSSIM (we refer to it from now on as TOSSIM or PowerTOSSIM, interchangeably) includes a model of the power consumption of the Mica2 nodes [5], but this model does not include the specific architecture of the node that we are studying and, more importantly, it does not take into account the important parameters that affect the power consumption. These parameters are analyzed in depth in the following section.

When we compile an application we have to choose the supported platform on which the executable file generated is going to run on. We have modified the simulator to make it able to imitate the behavior of the Sensor Cube platform. First, we have added more details to the simulation engine to correctly model the power factors explained in the next section. Then, a new model has been added to compile the applications for TOSSIM. This model accurately imitates the behavior of our node, as we show in our experimental results in Section 6. As a result, our extended model of the sensor nodes in PowerTOSSIM is much more precise and can be used to simulate the interaction of the processing and communication tasks in real-life nodes.

5. Power model

Analyzing the power consumption of the node, we have measured that the part of the nodes that consumes more power is the radio chip (typically between 70% and 90% of the overall dissipated power, depending on the application). Therefore, our extended power model treats in detail the power consumed by the radio.

The following sources are major causes of power waste in the radio:

Collisions: when two packets are transmitted at the same time and collide, they become corrupted and must be discarded and transmitted again. PowerTOSSIM models collisions as a logical or operation. Also there is no cancellation. This means that distance does not affect signal strength; if mote B is very close to mote A, it cannot cut through the signal from far-away mote C [15].

Idle listening: it happens when the radio is listening to the channel to receive possible data. The waste of power due to this factor can be very high, especially in applications where there is no data to send during the period when nothing is sensed.

Overhearing: this is produced when a node receives packets that have as destination one of the other nodes. The node that is overhearing consumes power receiving the packet and discarding it.

Control packet overhead: sending, receiving and listening for control packets consume power. Since control packets do not carry data, they reduce the throughput.

Current models for Wireless Sensor Networks do not take into account all these factors that affect the power consumption very significantly as we demonstrate in the case study. Therefore, we have added to our model all these factors.

Collisions are modelled by TOSSIM as a logical or, as we said before, then a node can receive corrupt packets. The receiving node checks the packet and discards it if it is corrupted or accepts it otherwise. The power consumed for idle listening is also taken into account by TOSSIM. Overhearing happens in our architecture, but in a different way. In the case of traditional radio transceivers, when a packet is received it is sent to the microcontroller, which is the component in charge of checking the address, CRC, etc. The Nordic nRF2401 checks the address and CRC before sending the packet to the microcontroller; this special behaviour is added to our model. The last important factor in the power consumption, control packet overhearing, is not covered by the current models. In fact, the original models contained in TOSSIM do not even consider acknowledgements, which are very influent in the power consumed by a network. A node sends an acknowledgement each time that it receives a packet, to communicate to the sending node that the packet is received correctly. TOSSIM guarantees that a bit sent is always received. However, if we implement a MAC protocol we need to use acknowledgments to simulate the behaviour of the network in order to obtain real values of power consumption. Only if the protocol is collision free, we do not need it to simulate the network behaviour.

The microcontroller consumes a very little amount of power; in fact, all the models overlook the microcontroller. We do not do a cycle-accurate simulation of the microcontroller, because it would unnecessarily increase the simulation time without a significant gain in precision, but we have included a good approximation based on the workload of the processor. For this, we count the number of microcontroller cycles that the were performed during the simulation and we multiply it by the average energy consumed by the microcontroller in a cycle. This coarse-grain analysis can be

done because the microcontroller consumes a very small amount of power when compared to the radio, for all possible types of instructions.

In order to perform the energy analysis, we need to determine the values of the power consumption for the considered components of the Sensor Cube hardware platform. These values have been measured during the operation of the physical node.

6. Case study

We use as case study a body area network running a medical application, which performs a remote, accurate, real-time analysis of the activity of the human heart. This has always been a challenging problem for biomedical engineers. Heart disorders like cardiovascular disease and stroke remain by far the leading cause of death in the world for both women and men of all ethnic backgrounds [10].

Our nodes are equipped with sensors that are able to perform *electrocardiogram/electroencephalogram (ECG/EEG)* analyses, which can readily reveal a number of heart malfunctions.

This kind of applications usually consists in a small number of nodes (between 5 and 10, depending on the application) with a star topology. The nodes are heterogeneous; some nodes can perform ECG, while others are performing EEG or additional tests. Radio transmissions are very short range because of the proximity between nodes. In order to perform a reliable ECG or EEG test, we need to sample at 1 KHz. For this reason the throughput of these networks is very high: we send 1 measurement/ms to the base station in the case of ECG and 24 measurements/ms in the case of EEG, since the EEG samples 24 channels at a time.

TinyOS was ported to the Sensor Cubes using the ShockBurst mode. The MAC protocol implemented originally was a very simple ALOHA with acknowledgements [9]. With this protocol, when a node has to send a packet it does not sample the medium to know if another node is transmitting at the same time. The probability of receiving corrupted packets with ALOHA is very high, especially for data intensive applications, such as ours. Due to the high throughput of the network it is highly probable to have collisions, which increases the power consumption.

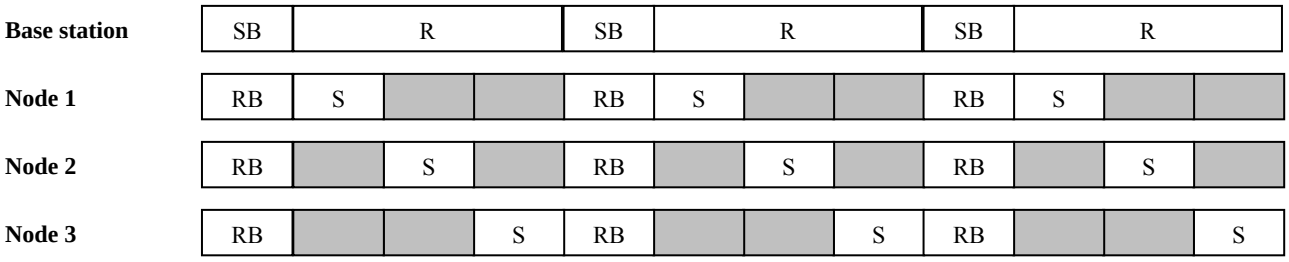

Figure 4. Scheme of operation of a TDMA protocol

Figure 5. Scheme of operation of the first implementation of our TDMA protocol

Figure 6. Scheme of operation of the second implementation of our TDMA protocol

In the case study we use our power model to compare different MAC protocols in terms of the power consumption. We have implemented a TDMA protocol to reduce power consumption and we improve this protocol making it dynamic w.r.t. the possibly changing number of nodes in the network. The simulations results of these two TDMA protocols are presented in this section.

TDMA (Time Division Multiple Access) is a MAC protocol which allows several users to share the same frequency channel by using different established transmission

timeslots. The users transmit in a rapid sequence, where each of them transmits using the assigned timeslot. These mechanisms enable multiple stations to share the same transmission medium (e.g. radio frequency channel), while using only the part of its bandwidth they require. Figure 4 represents the TDMA scheme for 3 nodes and a base station. Rectangles in grey mean that the radio is in standby mode, *SB = Sending Beacon, RB = Receiving Beacon, R = Receiving, S = Sending data*. As we can see, the base station sends beacons regularly to signal the beginning of a TDMA cycle to the nodes, the rest of the time it is listening to the possible transmissions from the nodes. Each node requests and it is assigned a timeslot during the discovery operation phase of the network in which the node can send data if required.

The first TDMA protocol implemented to test the reliability of our model has the same behaviour as the one described in the previous paragraph. At the beginning of the simulation, all the slots are free and every node has to send a slot request to the base station, the base station informs the nodes that ask for a slot in the following beacon. Each node sends the slot request when it is turned on, and resends the request after some time if the base station does not assign it a slot. Once a slot is assigned to the node, it transmits data to the base station in that slot and receives the acknowledgements of those transmissions in the beacons. If the acknowledgement corresponding to a packet transmission is not received after some time, the node resends the packet. In figure 5 we can see how the slots are assigned (*SSR = Sending Slot Request*). In the first TDMA cycle, the node 3 sends a slot request; the following beacon informs the node that the first slot has been assigned to it. In the second TDMA cycle, the same thing happens with node 1, which is informed in the following beacon about the slot assigned to it.

We modified this protocol making it more dynamic. The nodes have a short life time, compared with regular computers, for example. Therefore, a more dynamic protocol is needed in order to maximize the use of bandwidth, and therefore energy, when a node is dead. It can also be used to increase the number of nodes of the network, in the protocol previously described; we have to know in advance the maximum number of nodes to calculate the length of the TDMA cycle and the slots. The protocol that we describe now adapts itself dynamically to the number of nodes of the network. At the beginning, the base station sends a beacon and leaves an empty slot after the beacon transmission. This empty slot is used only for slot requests. Then, when a node wants to request a slot, it sends the request in that empty slot. Next, in the following beacon, the node is informed about the slot assigned to it. The slots dedicated for transmission are located after the empty slot. Figure 6 depicts how slots are assigned in the new implementation (ES = Empty Slot). In the second TDMA cycle, the node 2 sends a slot request; the following beacon informs it that the first slot has been assigned to it. The third TDMA cycle is longer, because there is a slot after the empty one dedicated to the node 2, in this TDMA cycle, the node 3 sends a slot request (also in the empty slot) and it is informed in the following beacon of its slot assignation. We see again that the length of the TDMA cycle has increased due to the assignment of another slot for the node 3. In this implementation it is quite probable that two nodes request a slot at the same time. If this happens, the packet received by the base station in the empty slot will be a corrupted packet and it will have to be discarded. We try to avoid this problem by randomly initializing a variable that indicates the number of cycles that the node has to wait before requesting the slot, this variable is decremented when a beacon is received and when its value is 0, the node requests a slot in the following empty slot. With this implementation it is easier to establish the topology, as we see in the experimental results, because the slots are assigned faster to the nodes than in the first implementation.

Table 1 contains values obtained with PowerTOSSIM using the model for the Sensor Cubes. We run simulations of 60 seconds. The simulated network consists of 6 elements (5 nodes and a base station). We chose this configuration of the network because it is the one in use for ECG applications. As benchmark we use the SurgeImec application [3]. In SurgeImec, each node takes sensor readings and forwards them to the base station. For the first implementation we chose a TDMA cycle of 6 slots (one for the beacon and five for the nodes).

Table 1. Energy consumption for the two different versions of the TDMA protocol. The application was executed for 60 seconds and all values are in milijoules (mJ).

TDMA protocol	Base station	Nodes (average)
First implementation	2728.48	1121.84
Second implementation	2432.36	700.52

Table 2. Simulated component energy breakdown. All values are in milijoules (mJ) consumed in 60s.

	CPU	Radio	ADC	Leds	Total
Base station (first implementation)	143.60	2584.88	0.00	0.00	2728.48
Nodes (first implementation) – average	137.58	732.16	87.71	164.37	1121.84
Base station (second implement) – average	143.91	2288.44	0.00	0.00	2432.36
Nodes (second implementation) – average	138.11	427.08	85.82	54.92	700.52

In the results showed in table 1 we can appreciate that the energy consumed by the base station with the first implementation is slightly higher than with the second one (13% less energy in the latter case). This is due to the fact that the base station in the first implementation uses one slot in TX mode and 5 slots in RX mode, while the second implementation adapts the TDMA cycle length to the nodes that are ON. Moreover, especially at the beginning, the power consumption is lower because the proportion of time that the radio is listening (the mode in which the radio consumes more power) is lower. Similarly, the power consumed by the nodes is significantly lower in the second implementation (37% less energy consumption), which is basically due to the reduction in the number of collisions when different nodes request a slot.

In table 2 we show a more detailed analysis per component of the overall energy values consumed in the nodes and base station. This table indicates that the energy consumed by the CPU is almost the same in all the cases. However, the radio has a very different impact for the base station and the nodes. As it has been initially designed with this purpose, the nodes' radio consume significantly less energy than the radio of the base stations (2x less), illustrating the benefit of the power saving policy implemented in the TDMA of the nodes. In this case, the radio is on only when it sends a packet or when it is waiting to listen for the beacon, and the rest of the time our implementation keeps the radio off since it is not needed to transmit any biomedical data.

7. Conclusions

Nowadays, WSNs are growing in size, complexity and fields of application. Therefore, it is becoming more difficult to debug applications, test new improvements in the nodes, etc. The experience has proven that flexible WSN simulators are essential to achieve these goals and make easier the evaluation of different WSN configurations (e.g., topologies, communication protocols, etc.) according to the underlying type of monitoring. In addition, reliable and accurate node models for these simulators are required to accurately simulate the behavior of our target architecture without waiting for the final implementation of the nodes.

In this paper, we have proposed a precise model for power assessment in WSNs, which can achieve the goals previously mentioned. All the important parameters that can affect the power consumed in a WSN are considered by the proposed new model. We have validated the proposed methodology by accurately modeling a real-life sensor nodes and base station, and assessing the energy figures of two TDMA MAC protocols in a WSN.

8. Acknowledgements

This work is partially supported by the Spanish Government Research Grant TIN2005-5619 and the Swiss FNS Research Grant 20021-109450/1.

9. References

[1] Culler, D., et al. Overview of Sensor Networks. Computer, August 2004, 41-49.

[2] Ye, W., Heidemann, J. Medium Access Control in Wireless Sensor Networks. USC/ISI Tech. Report ISI-TR-580, 2003.

[3] Torfs, T.,et al. Wireless network of autonomous environmental sensors. 3rd IEEE Conf. on Sensors, 2004.

[4] Levis, P., et al. TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications. ACM Sensys, 2003.

[5] Shnayder, V., et al. Simulating the Power Consumption of Large-Scale Sensor Networks Applicats. ACM Sensys, 2004.

[6] Culler, D. TinyOS: Operating System Design for Wireless Sensor Networks, Sensors, May 2006.

[7] Texas Instruments MSP430 Microcontroller. www.ti.com

[8] Nordic nRF2401 Radio chip. www.nordicsemi.no

[9] Daskalopoulos, I.,et al. Power-efficient and Reliable MAC for Routing in Wireless Sensor Networks. MSc thesis, University College London, 2005.

[10] Fuster, V. Epidemic of Cardiovascular Disease and Stroke: The 3 Main Challenges. Circulation, Vol. 99(9), 1999.

[11] Paradiso, J.A.,et al. Energy scavenging for mobile and wireless electronics. IEEE Pervasive Computing, 2005.

[12] Gay, D., et al. The nesC language: A holistic approach to networked embedded systems. Proc. PLDI, June 2003.

[13] Lo, B., et al.Body Sensor Network-Wireless Sensor Platform for Pervasive Healthcare Monitor. Proc. PERVASIVE, 2005.

[14] Feng, J., et al.. System-architectures for WSN issues, alternatives, and directions. IEEE ICCD, pp. 221-226, 2002.

[15] Philip Levis and Nelson Lee. "TOSSIM: A Simulator for TinyOS Networks"