OS-Based Sensor Node Platform and Energy Estimation Model for Health-Care Wireless Sensor Networks

omitted for blind review

Abstract

Accurate power and performance figures are critical to assess the effective design of possible sensor node architectures in Wireless Sensor Network (WSNs) since they operate on limited energy storage. Therefore, accurate power models and simulation tools that can model real-life working conditions need to be developed and validated with real platforms. In this paper, we propose a sensor node platform designed for health-care applications and a validated simulation model based on event-driven operating system simulation that can be used to accurately analyze performance and power consumption in healt-care WSNs composed of multiple nodes. Thus, this model can be employed to tune the node architecture and communication layer for different working conditions of applications and topologies of WSNs. In this paper we illustrate how the proposed simulation model can be used to tune the Medium Access Control (MAC) layer in the architecture of the proposed sensor node to optimally use it for different applications and working conditions. Our results show variations of less than XXX% between the presented simulation framework and measurements in the final platforms.

1 Introduction

Wireless Sensor Networks (WSN) [3] consist of a number of spatially-distributed sensor nodes connected through a wireless network to cooperatively monitor physical magnitudes. The main task of WSNs entails sensing relevant information in the environment and transmitting it to a base station, which store and process the data. Latest enhancements in ultra-low-power processing architectures [11] and scavenging mechanisms [14, 9] have enabled the design of low-cost, small and lightweight sensor nodes; thus, WSNs have enlarged their possible range of applications, including nowadays, environmental measurements or biomedical monitoring [6].

In all possible working environments, one of the most critical design constraints to successfully deploy WSNs is power consumption [1]. Since WSNs should autonomously work and avoid maintenance, they need to operate and self-sustain with very limited energy resources, such as, batteries or energy scavengers (e.g., solar cells). Moreover, in occasions the nodes need to function in hostile or non-easily accessible areas, which makes it very difficult (or even impossible) to replace the battery. In fact, health-care applications of WSNs or *Body Area Networks (BANs)* are one of the most challenging cases, because replacement of power supplies in patients is a very tedious and unpleasant task, specially if the nodes are implanted in the body. Moreover, the type of monitoring that needs to be performed in each pacient can be significantly different. Therefore, development of ultra-low-power and versatile sensor node architectures that can be configured for different types of health-care monitoring are key for the final success of WSNs in this context.

The development of flexible sensor nodes for BANs implies challenges at various design levels, as tuning sensor node architectures include possible optimizations at different layers of abstraction, namely, hardware, application software and *Operating System (OS)*-middleware (e.g., *Media Access Control (MAC)*, task scheduler, etc.). Moreover, optimal values of individual node parameters are heavily affected by network-level parameters [6], such as, network topology or total number of active nodes in the network. Hence, sensor node architecture can only be correctly developed and tuned for each working environment when complete WSN can be modelled and accurate power and performance figures can be extracted. As a consequence, the development of power and performance estimation frameworks of complete topologies for BANs in real-life working environment, which incur only in a limited and predictable error for manufactured sensor node architectures are in great need.

In this paper we propose as a versatile OS-based sensor node platform targetting ultra-low-power health-care BANs. It includes a porting of the TinyOS operating system [2] to enable a fast reconfiguration of the type of biomedical application to be executed in each working environment. Moreover, to enable a larger efficiency in the architectural tuning process of the node for different configurations of BANs, we also present a complete performance/power simulation framework that models the interaction of the different layers of its architecture (i.e., hardware, OS and application). This framework relies on TOSSIM [12], an event-driven TinyOS simulator, to accurately model the functionality of the microcontroller, the radio device and communication between nodes at the physical level of WSNs. Our results show the proposed framework achieves power and power estimations of real-life BANs made of our OS-based sensor nodes with differences of less than XXX%. Thus, our framework improves the accuracy of PowerTOSSIM [19], the recently developed extension of Tossim for power estimation of WSNs by XXX%.

The rest of the paper is organized as follows. In Section 2, we overview related work in WSN design and modelling. In Section 3, we describe the hardware and software architecture of our health-care sensor node. The proposed OS-based performance/power modelling framework is presented in detail in Section 4. Then, in Section 5 we assess the accuracy of the proposed model for our platform to tune the MAC layer to minimize power dissipation on real-life working conditions of health-care monitoring. Finally, in Section 6 we summarize our conclusions and discuss future research lines.

2 Related Work - To modify: HOLST

The study of Wireless Sensor Network has attracted several research groups. One of the most active belongs to University of California, Berkeley. They are responsible for designing platforms like Mica [13] or Telos [16] motes. Moreover, they have carried out some other well know projects like the development of a very extended operating system for WSN, TinyOS [2], and its simulator, TOSSIM [12]. NesC [7] programming language was also developed by this group, it is an extension of C language designed to embody the structuring concepts and execution model of TinyOS. In addition, PowerTOSSIM [19] is a power modeling extension to TOSSIM. It provides an accurate and fast per-node estimate of power consumption. PowerTOSSIM employs a code-transformation technique to estimate the number of CPU cycles by counting the execution of basic blocks and mapping them to clock cycles of the microcontroller. This theoretically allows to obtain a detailed analysis of the energy spent. However, several issues affect the accuracy of PowerTOSSIM for real-life WSNs and BANs. First, it needs an accurate mapping from the basic blocks to binaries, since the modeling is based on basic block level. Second, some low level components and network communication effects (e.g., collisions, distance of the nodes, etc.) are ignored or significantly simplified during the mapping and simulation of the complete WSN, which affects the accuracy of the estimations, especially for the radio, as we show in this paper for our propose node architecture and different BAN configurations.

Another important project in WSN is SmartDust [20], where a hypothetical network of tiny wireless microelectromechanical systems (MEMS) sensors, robots, or devices, provided with wireless communications, can detect anything from light and temperature, to vibrations, etc. A very recent work is NEST [18], an Open Experimental software/hardware Platform for Network Embedded Systems Technology research that is expected to accelerate the development of algorithms and services to compose complex monitoring applications.

Harvard University is developing CodeBlue [17], it consists of a set of devices that collect heart rate (HR), oxygen saturation (SpO2), and EKG data and relay it over a short-range (100m) wireless network to any number of receiving devices, including PDAs, laptops, or ambulance-based terminals. The data can be displayed in real time and integrated into the developing pre-hospital patient care record. The sensor devices themselves can be programmed to process the vital sign data, for example, to raise an alert condition when vital signs fall outside of normal parameters. Any adverse change in patient status can then be signaled to a nearby EMT or paramedic. The device used has been also developed by this group, and consists in a wearable wireless pulse oximeter and 2-lead EKG based on the Mica2, MicaZ, and Telos sensor node platforms.
Apart of the development of sensor, there are groups devoted to the study of operating systems for WSN. Examples of Operating System are Conti6kiOS [5], SOS [8] or RTOS [4].

As was discussed in the introduction the energy consumption is one of the main issues in WSN, because their size does not allow a large batteries, and their location, sometimes in inaccessible areas, does not let change the battery easily. However, in the literature is only present a part of the energy model. For example, [10] is centered in the energy consumption of the sensor's processor, whereas others authors are only worried about the energy consumption of the wireless data transmission [22] and [21].

3 Health-Care Sensor Node Architecture

The energy model that we developed is based on a platform called Wireless 24+1 channel EEG/ECG system designed by IMEC-NL. This small ultra-low-power sensor node is capable of monitoring up to 24 channels Electroencephalogram (EEG) and 1 channel Electrocardiogram (ECG), and it is widely used in the emerging technology of body area networks (BAN) [15].

3.1 Hardware Architecture

Figure 1 depicts the high level architecture of the system. The platform is partitioned in three main blocks which perform the main operations: sensing, processing and transmitting the data to a collecting device (PC or PDA). The sensing part relies on a 25 channels ultra-low-power ASIC (application specific integrated circuit) to extract the bio-potential signals. Adjustable gain and bandwidth allow to easily switch between ECG and EEG monitoring, in order to fit a wide range of applications. The processing capability is provided by a TI MSP430 ultra-low-power microcontroller. This small 16-bit RISC microcontroller has a very low active power (0.6 nJ/instruction), low stand-by power (2μ W), fast wakeup from stand-by to active mode (6μ s) and an on-chip 12-bit analog-to-digital converter. It also includes 60kB of flash program memory and 2kB of RAM.

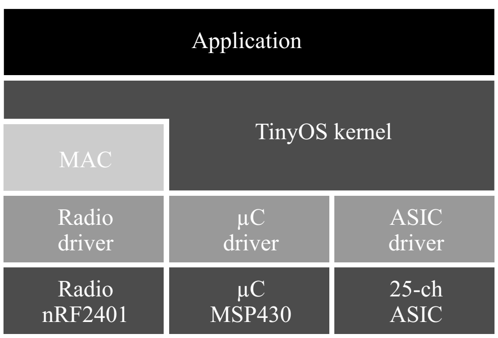

Figure 1. OS-Based EEG/ECG 24+1ch sensor node architecture overview

The wireless communication is based on the Nordic nRF2401, an ultra-low-power single-chip 2.4 GHz transceiver. This component has the lowest power consumption in its category, with only 10.5mA at an output power of -5dBm and 18mA in receive mode. Built-in power down modes allow to switch-off the radio when not used. This radios has two modes of operation: Direct and ShockBurst. In Direct mode the radio works like a traditional RF device. Data must be at 1Mbps 200ppm, or 250Kbps 200ppm at low data rate setting, for the receiver to detect the signals. The ShockBusrt technology uses an on-chip FIFO to clock in data at a low data rate and transmit at a very high rate thus enabling extremely power reduction. This mode of operation reduces the consumption. We can use a low frequency crystal to clock in data but the transmission rate is very high. Other 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 receiving packet.

3.2 Software Architecture - To modify: HOLST

The software architecture of the platform has been specifically designed to match the simulation environment. We therefore chose a layered modular approach in which each platform component is a separate software block. This allows to easily replace hardware related blocks (e.g. radio, microprocessor, etc.) with their correspondent models in the simulator without modifying the upper layers (communication protocols, application, etc.). The node includes an embedded operating system, TinyOS, on top of which the entire software architecture is built.

3.2.1 TinyOS

TinyOS is an embedded operating system for wireless sensor networks. Its small footprint allows it to be coupled with ultra-low-power microcontrollers, which usually have very small embedded memories. While a porting for the TI MSP430 microcontroller is already available within the TinyOS community, new drivers have been developed to port the radio module and the 25 channel EEG/ECG ASIC to the operating system.

3.2.2 Communication protocols - To modify: UCM

We have implement MAC protocols in order to compare them in terms of energy. We have chosen the TDMA protocol (Time Division Multiple Access), which allows several users to share the same frequency channel by dividing the signal into different time-slots. This lets multiple stations to share the same transmission medium (e.g. radio frequency channel). ***(ADD FIGURE about static T-DMA and Dynamic T-DMA (see ppt in the Wiki))***. The base station sends beacons regularly to signal the beginning of a TDMA cycle to the nodes. Each node has a slot assigned, if the node has data to send, it would be sent in that slot.

A static TDMA protocol has been implemented. At the beginning of that protocol all the slots are free and every node has to send a slot-request to the base station. The base station answers the nodes that have demanded a slot in the following beacon. Then, each node sends a slot-request when it is turned on, and resends it after some time if the base station has not assigned any slot to the node. Once the slot is assigned, the node 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 this case the number of slot is fix and known in advance. In figure X we can see how the slots are assigned. In the first TDMA cycle, the node *i* sends a slot request, the following beacon informs it that the first slot has been assigned to it. In the second TDMA cycle, the same thing happens with node *j*.

We have improved this protocol because it wastes bandwidth and so energy. The nodes have a short life time, compared with regular computers, then, the number of nodes can clearly decrease during the time. A dynamic TDMA protocol is proposed in order to adapt 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. When a node wants to request a slot, it sends the request in that empty slot. After, in the following beacon, the node is informed about the slot assigned to it. The empty slot is always the one after the beacon and the slots dedicated for transmission are located after the empty slot. The size of the TDMA cycle depends on the number of nodes that have request a slot. If two nodes request a slot at time, the packet received by the base station in the empty slot will be corrupted and is discarded. The protocol avoids this problem by a random variable that indicates the number of cycles that the node has to wait before requesting a slot. Figure Y depicts how slots are assigned in the new implementation. In the second TDMA cycle, the node i 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 i. The rest of the nodes proceed in the same way.

4 Energy modelling framework - To modify: UCM

There are two main sources of energy consumption in the node, the microcontroller and the radio. Measuring the node energy consumption the radio means typically between 90% and 50% of the total energy consumption, depending on the application and the radio operation mode. Thus, an accurate energy model of the radio is mandatory for an effective analysis of the whole platform, although we have also modeled the microcontroller.

We have located the major causes of energy waste in the radio, modeled it, and implemented it using TOSSIM [12] to make it able to imitate the behavior of our platform. We add more detail to the IMEC model to simulate all the major causes of energy waste. In this implementation we insist on an accurate modelling of the physical channels and the possible conflits between different nodes that try to trasmit data at the same time. Then a new model, more accurate than the previous one, has been added to the platform ”pc”, that is the one used to compile the applications for TOSSIM.

In the following subsections we describe separately the two main components modelled: the microcontroller and the radio.

4.1 Microcontroller Modelling

The microcontroller is supposed to consume a very little amount of energy; in fact, in all the models the microcontroller is discarded. In ours, the microcontroller is not simulated at the bit level, because it would increase too much the simulation time, but we make a very good approximation that results of multiplying the number of cycles that the simulation takes by the average power consumed by the microcontroller in a cycle. In the nodes that we are using, the microcontroller is always in active mode, this is required by ASIC's driver, since it needs a synchronization signal to get data from the sensors that has to be provided by the microcontroller. Then we calculate the energy consumption applying $E = IV_{dd}t_{cycle}$. We have measured the node and we have obtained that, in active mode, the mean current that circulates through the microcontroller is 2.152mA, while the current drops to almost 0mA when the microcontroller is switched to sleep mode. For the measurements, we used a 2.8V power supply.

4.2 Radio and Physical Communication Modelling

The other component modeled is the radio. The radio chip inserted in our sensor node has two modes of operation: Direct and ShockBurst, as was described above. In our energy simulation the mode modeled is ShockBurst because it is the only mode included in the real platform used as case study. We have also measured the radio energy consumption through the radio current that circulates through it. When it is in receiving mode, the current is 22.88mA. It is 13.72mA in transmission mode and 0mA in standby. A 2.8V power supply was also used in this case.
Once determined the radio mode, we have studied the major causes of energy 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. Collisions are modeled by TOSSIM as a logical or, then a node can receive corrupt packets, that should be sent again in order to receive the correct packet, this will imply an increase in the energy consumption. However, in the TOSSIM model, it was assumed that all the packets reach the destination without errors, it was absolutely impossible to detect whether a collision had happened. In our case, collisions can be detected by the simulator in the same way as they are detected in the real hardware. When a node wants to send a packet, the microcontroller of the node sends the payload to the Nordic nRF2401, the radio chip adds a preamble, the destination address and CRC. When the packet is received by a node, it checks the packet integrity using the CRC added by the sending node and it discards the packet if

there is any error.
Idle listening: it happens when the radio is listening to the channel to receive possible data. The waste of energy 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. This work is centered on biomedical applications, which are not constantly active, but they are most of the time without transmitting any data to the base station. For this reason, idle listening has a very strong impact in the energy consumed by the radio in the applications that we are considering. The power consumed for idle listening is taken into account by TOSSIM, we did not improve the simulator in this aspect, because it is accurate enough regarding this factor.
Overhearing: this is produced when a node receives packets that are destined to other nodes. The node that is overhearing consumes energy receiving the packet and discarding it. Usually when a packet is received it is sent to the microcontroller, which is the component in charge of checking the address, CRC, etc. But the Nordic nRF2401 checks the address and CRC before sending the packet to the microcontroller, in the same way as it is done for collisions. This special behavior is also added to our model. When a node receives a packet, it checks the destination address included by the sending node. If both the packet address and the receiving node address are the same, the packet is given to the microcontroller. On the contrary, if both addresses are different, the packet is discarded and the microcontroller will not notice the reception of that packet.
Control packet overhead: sending, receiving and listening for control packets consume energy. Since control packets do not carry data, they reduce the throughput. Depending on some factors of the modelling (like the MAC or routing protocol used in the model, the use of acknowledgements (ACKs), etc.), the energy consumption can vary to a large extent. We include in our model all the control packets that are transmitted in our node (beacons, ACKs and slot requests), so we can estimate accurately the energy consumption of our sensor nodes using the simulator.

5 Case studies - To modify: HOLST

As outlined in the introduction, we chose as a test case the design of a small WSN for biomedical applications. The purpose of these kind of WSNs, also referred as Body Area Networks (BAN), is to monitor a wide range of bio-potentials signals, such as electrocardiogram (ECG), electromyogram (EMG), electroencephalogram (EEG) and electrooculogram (EOG). The BAN that we considered is composed by 5 different sensor nodes, each one responsible for the monitoring of a different signal. Of the 5 nodes, we focused on the ECG node, as ECG is one of the most common and more widely analyzed signals.
In order to validate our energy model, we tested its accuracy varying the communication protocol (MAC) and the application running over the sensor nodes. These two parameters strongly affect power consumption; therefore they are good candidates to prove the accuracy of the power estimations of our simulator.

5.1 ECG streaming application

Electrocardiogram (ECG) monitoring is certainly one of the most interesting applications in the field of biomedical monitoring. A huge amount of information can in fact be extracted from the ECG signal, such as heart rate, heart rate variability, etc.
In the first test application we developed, a 2 channels ECG signal is extracted from a node placed on the body and transmitted to the base station. The base station receives then a streaming ECG data, which can be processed to obtain all the needed information.

5.1.1 Static TDMA

5.1.2 Dynamic TDMA

5.2 Rpeak application

In the second test application we added a local preprocessing phase in the sensor node so that the load on the radio channel, and consequently the power consumption, can be reduced.
The Rpeak application detects when a heart beat happens in any of the channels. The main program takes samples from each channel at 200Hz. For each sample, it calls an algorithm that returns 0 if the current sample is not a beat. Otherwise, it returns a positive value that indicates how many samples ago a beat was detected in that channel. For example, if it returns 74, it means that the sample that we gave to the algorithm to be processed 74 calls ago was a beat. In this case, we can extract that a heart beat occurred 74*5ms = 370ms ago, one sample is taken every 5 ms. When the algorithm returns a value different from 0, a packet is sent to the base station to inform it about the detected beat.

5.2.1 Static TDMA

5.2.2 Dynamic TDMA

6 Conclusions - To be completed: At the end

References

[1] X. Cheng, B. Narahari, R. Simha, M. X. Cheng, and D. Liu. Strong minimum energy topology in wireless sensor networks: Np-completeness and heuristics. *IEEE Transactions on Mobile Computing*, 02(3):248–256, 2003.

[2] D. Culler. Tinyos: Operating system design for wireless sensor networks. *Sensors*, pages 41–49, May 2006.

[3] D. Culler, D. Estin, and M. Srivastava. Overview of sensor networks. *Computer*, pages 41–49, August 2004.

[4] I. Daskalopoulos, H. Diall, and K. Raja. Power-efficient and reliable mac for routing in wireless sensor networks. *Master's thesis, , University College London*, 2005.

[5] A. Dunkels, B. Grnvall, and T. Voigt. Contiki - a lightweight and flexible operating system for tiny networked sensors. In *Proceedings of Proceedings of the First IEEE Workshop on Embedded Networked Sensors*, 2004.

[6] J. Feng, F. Koushanfar, and M. Potkonjak. System-architectures for sensor networks issues, alternatives, and directions. *ICCD*, 00:226, 2002.

[7] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler. The nesc language: A holistic approach to networked embedded systems. In *PLDI'03:Programming Language Design and Implementation*, June 2003.

[8] C. Han, R. Kumar, R. Shea, E. Kohler, and M. Srivastava. A dynamic operating system for sensor nodes. In *MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 163–17, 2005.

[9] J. Hsu, J. Friedman, V. Raghunathan, A. Kansal, and M. B. Srivastava. Heliomote: Enabling self-sustained wireless sensor networks through solar energy harvesting. In *ACM-IEEE International Symposium on Low Power Electronic Design (ISLPED)*, 2005.

[10] G. Kk and C. Basaran. Reducing energy consumption of wireless sensor networks through processor optimizations. *JOURNAL OF COMPUTERS,*, 2(5):67–74, July 2007.

[11] J. Lach, D. Evans, J. McCune, and J. Brandon. Power-efficient adaptable wireless sensor networks. In *International Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*, 2003.

[12] P. Levis, N. Lee, M. Welsh, and D. Culler. Tossim: Accurate and scalable simulation of entire tinyos applications. In *SenSys*, August 2003.

[13] N. mote. http://www.xbow.com.

[14] J. A. Paradiso and T. Starner. Energy scavenging for mobile and wireless electronics. *Pervasive Computing, IEEE*, 4(1):18–27, 2005.

[15] J. Penders, B. Gyselinckx, N. de Vicq, and T. Torfs. Body area networks for multi-modal biomedical monitoring. In *pHealth*, June 2007.

[16] J. Polastre, R. Szewczyk, and D. Culler. Telos: Enabling ultra-low power wireless research. *Fourth International Symposium on Information Processing in Sensor Networks (IPSN)*, pages 364–369, April 2005.

[17] C. B. Project. http://www.eecs.harvard.edu/ mdw/proj/codeblue/.

[18] N. project. http://nest.cs.berkeley.edu/.

[19] V. Shnayder, M. Hempstead, B. Chen, G. Allen, and M. Welsh. Simulating the power consumption of large-scale sensor network applications. In *SenSys*, 2004.

[20] B. Warneke, M. Last, B. Leibowitz, and K. Pister. Smart dust: Communicating with a cubic-millimeter computer. *Computer*, pages 44–51, January 2001.

[21] L. Zhong. A unified data-link energy model for wireless sensor networks. In *PhD Thesis - University of California Berkeley*, pages 212–227, 2007.

[22] B. Zurita, C. Fischione, and K. H. Johansson. Energy consumption of minimum energy coding in cdma wireless sensor networks. In *EWSN*, pages 212–227, 2007.