International Journal of
Sensor Networks and Data Communications

Research Article Open Access

A Rapid Prototyping Matlab Based Design Tool of Wireless Sensor Nodes for Healthcare Applications

Rammouz R[1,2]*, Labrak L[1], Abouchi N[1], Constantin J[2], Zaatar Y[2] and Zaouk D[2]

[1]Nanotechnology Institute of Lyon, Lyon, France
[2]Applied Physics Laboratory, Beirut, Lebanon

Abstract

Remote patient monitoring enables medical facilities and personnel to monitor patients outside of conventional clinical settings. This technology is greatly affected by the available energy sources. The limited energy supply is the bottleneck for measurement, data transmission, network connection and lifetime. Thus, efficient power consumption estimation in the early stage of the design would give the user an insight on its feasibility within medical constraints. In this paper, a power-oriented Matlab based design tool is proposed. It will help the user to make an energy aware design. In fact, it will enable him to determine the power consumption of each node based on a set of selected components, the daily routine of the patient and many other factors. It will also allow him to test multiple configurations prior to the physical implementation. This will minimize the time and financial impact of any modification. Furthermore, it will provide guidance to properly adapt the design within the requirements of the application: Choosing components, energy sources, etc. A validation is proposed via a physical implementation in order to compare the results obtained through simulation with practical measurements. Future works aim to take into account memory access operations, include several sensors operating within a network, and adapt the tool to other applications (smart buildings, environmental monitoring, etc.).

Keywords: Remote patient monitoring; Power consumption; Matlab; Energy aware design; Validation

Introduction

The ever increasing number of people who need continuous medical attention coupled with the rising costs of healthcare has triggered the concept of remote patient monitoring. This can be achieved using wireless sensor nodes attached to the human body. These nodes ensure periodic measurements of vital signs [1] such as heart rate, blood pressure, temperature, etc. Data acquired is first sent to a local gateway (cellphone, tablet, PDA …) through short distance wireless protocols. It is then uploaded to a medical database, analyzed and viewed by healthcare professionals (Figure 1) [2,3].

Remote patient monitoring is frequently used with the elderly and the chronically ill. It allows these patients and their physicians to closely monitor their medical condition and, if needed, intervene [4]. This technology is mainly limited by the amount of available energy. Thus estimating the power consumption of a node prior to its physical implementation will allow the designer to properly adapt its architecture (choosing energy sources, selecting components, etc.).

Some ready to use network simulators already have energy profiling functions. Nevertheless none of these can guide a designer in the decision process at electrical architectural level. The developed tool is power oriented and has two advantages. The main is an easy configurable component based environment. It allows fast creation of new component description based on the datasheets thus producing a flexible growing database. The second lies in many built-in solutions coupled with optimization processes that provide the user with guidance to properly adapt his design within the constraints of the application: choosing the more power efficient configuration, the required energy sources, etc.

The rest of the paper is organized as follows. Section 2 gives an overview on energy profiling and networked embedded systems simulators. In section 3, we will explain the methodology used to determine the power consumption of the node and implementation of the built-in functions. We will then present its application to a wireless temperature sensor node in section 4. Section 5 will be dedicated to the physical implementation of that node. The results are then presented in section 6 and the conclusion is in section 7.

State of Art

Energy profiling has already been at the center of several studies.

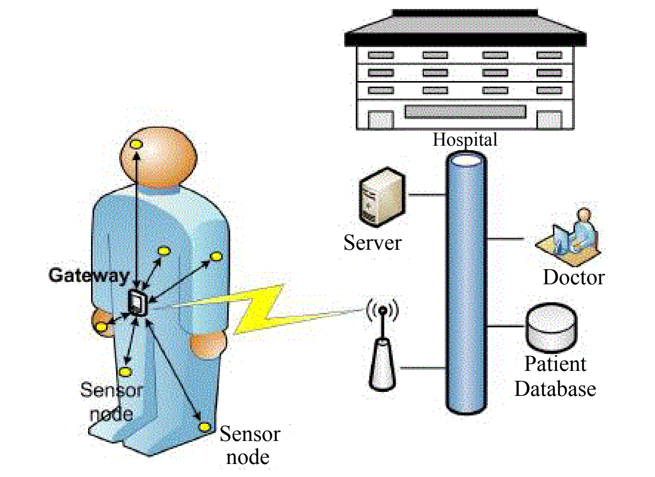

Figure 1: A remote patient monitoring system’s architecture.

***Corresponding author:** Rammouz R, Nanotechnology Institute of Lyon, Lyon, France, Tel: (33) 6 46 21 88 54; E-mail: ramzy.rammouz@cpe.fr

Received June 30, 2016; **Accepted** July 18, 2016; **Published** July 28, 2016

Citation: Rammouz R, Labrak L, Abouchi N, Constantin J, Zaatar Y, et al. (2016) A Rapid Prototyping Matlab Based Design Tool of Wireless Sensor Nodes for Healthcare Applications. Int J Sens Netw Data Commun 5: 144. doi: 10.4172/2090-4886.1000144

Plenty of networked embedded systems simulators have been developed in the last decade. This is mostly due to the uncertainty faced by designers regarding the system's energy consumption.

A first example is the general network simulators such as NS-2, NS-3 [5] and OPNET [6]. These simulators lack in the infrastructure for easy and accurate hardware modelling despite having an important amount of protocol implementations.

A second example is specific ad hoc networks simulators. Those were created to overcome general network simulators deficiencies regarding system models for simulating sensor nodes. They included some specific models for radio propagation, sensors and hardware systems. SENSE (SEnsor Network Simulator and Emulator) is one of these simulators. It is built over a component based modelling approach. Yet it lacks in the implementation of commonly used wireless standards and does not include models for noise or interference [7].

A third example is that of virtual prototyping as in IDEA1. IDEA1 is based on System C and C++ language. It allows cycle-accurate energy consumption estimation. Moreover, it has an instruction set simulator that supports different microcontrollers [8]. However, accurate models require disclosure of details that are not always provided by microcontroller designers [9].

A last example is the multi-domain simulation based on simulators coupling. Interfacing simulators is a complex task taking into account that each is developed in a different programming language and has its own user interface.

These factors lead us to propose a new approach: creating a design tool that is generic enough to make it simple to implement new components without a loss of accuracy. This will significantly speed up the production cycle especially when designers have prior knowledge of each component's individual performances.

Design Tool

In this section we aim to explain the concept we were based on to determine the power consumption of the node.

Only then, we will be able to elaborate on the built-in functions.

Concept

Our goal is to simulate a wireless sensor node and estimate its power consumption taking into account a physical implementation. The first step was to select an efficient modeling approach. We used a Model Based Design (MBD) approach as it brings cost and time saving in the development. Moreover the function models can be easily reused for different setups. MATLAB/SIMULINK offers an adapted environment and many suitable features to implement our system as an executable specification.

Our purpose is to determine the power consumption of a wireless sensor node similar to the one shown in Figure 2. In order to do that, we need to identify the activity of each component. It is obtained from the time spent in each state (active, sleep, idle …) as well as state transitions [10]. This conducted us to establish the cycle of operation presented in Figure 3 [11] and explained here under:

- The processor wakes up periodically in accordance with a predefined measurements rate following an assessment of the patient's medical condition
- The processor activates the sensor to start converting the physiological signal into an electrical one
- Simultaneously, an analog to digital conversion is taking place in the ADC
- Once the measurement/conversion process is done, both the sensor and the ADC go to sleep mode
- The microprocessor processes the ADC output and data is sent to the RF transceiver in order to be transmitted to the gateway
- Meanwhile the processor becomes idle: the CPU is disabled during transmission, and the processor waits for an interruption from the wireless module
- The RF transceiver informs the processor of the outcome of the transmission and is put to sleep
- At the end, the whole node goes back to sleep mode

Based on this cycle, we have modeled the instantaneous current consumption of each component as well as the whole unit using Matlab. Figure 4 shows the blocks embedded in our model. Component and configuration related inputs are used to extract consumption waveforms as well as mean values [11].

Built-in functions

In order to provide an efficient reconfigurable tool, we decided to adopt a model based approach. It consists in two simple built-in functions coded in Matlab. The first one computes current consumption of a given node, while the other provides the user with guidance during the design process.

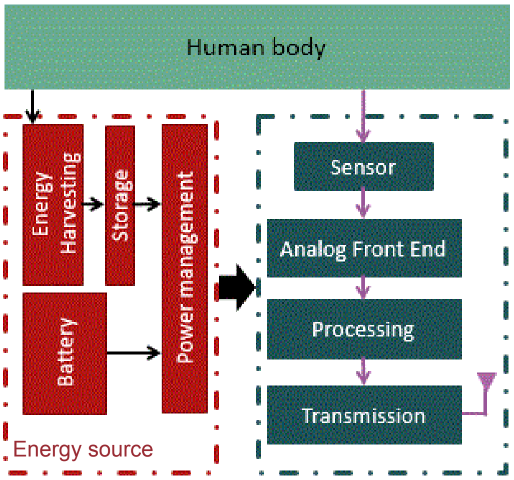

Figure 2: Block diagram of a wireless sensor node.

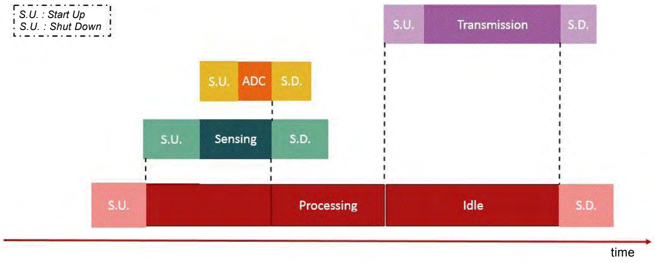

Figure 3: Cycle of operation of the sensor node.

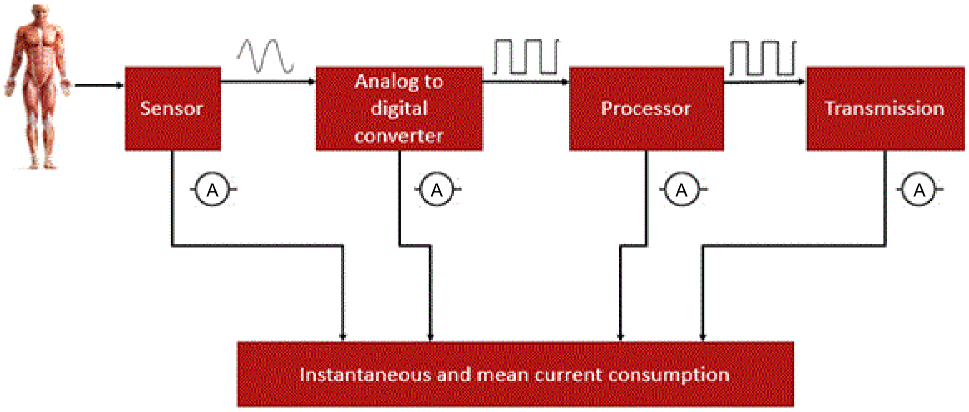
Sensor
Analog to digital converter
Processor
Transmission
A
A
A
A
Instantaneous and mean current consumption

Figure 4: Blocks embedded in the model with their respective inputs/outputs.

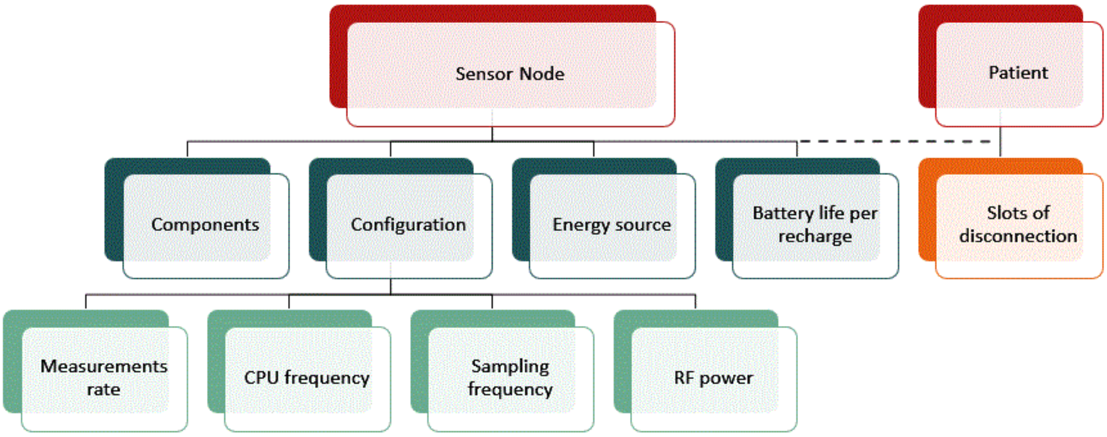

Figure 5: Variables in a wireless sensor node for medical applications.

Variables / Scenario	Battery life	Disconnection slots	Configuration of the node				
			Energy source	Measurements rate	Sampling frequency	CPU frequency	RF power
Battery life per recharge	🟥						
Picking energy sources			🟥				
Disconnection time slots		🟥					
Design			🟥	🟥	🟥	🟥	🟥

Legend: 🟥 Variable □ Parameter

Table 1: Variables/parameters combination for each scenario.

Estimating the current consumption of the node: Our goal is to determine for a given node, current consumption throughout the whole monitoring period which extends to weeks, maybe months. This consumption is sometimes affected by the outcome of the connection to the gateway. In fact, the RF module's operation may differ whenever we are connected or not. This is the case for some RF communication standards such as ZigBee [12], Bluetooth [13], and Bluetooth Low Energy [14]. Since these protocols are the most used in sensor network, calculating the current consumption of a node throughout the whole monitoring period has to take into account any disconnection imposed by the daily routine of the patient. And that's exactly what the first built-in function does.

Guidance towards an energy aware design: Another built-in function aims to guide the user to a power efficient design within the medical requirements of the application. It computes optimization routines to determine the best trade-off among several possible configurations. The diagram shown in Figure 5 displays the variables in a wireless sensor node. We can classify these variables in two main categories: patient related or node related. While designing a node, some of these variables can be set as parameters according to feasibility or medical constraints. This function includes solutions for four different scenarios. Each scenario represents a different configuration of these variables i.e., a different variables/parameters combination (Table 1).

Battery life per recharge: In this case, the architecture of the node is fixed, the energy source is chosen, and the patient has an established daily routine (disconnection slots). We aim to extract the mean current consumption of the node and determine, for a given battery, how long it will keep the sensor node operating without needing a recharge.

Picking energy source: This scenario assumes that a minimal battery life is required. Moreover, disconnection time slots are already established and the architecture of the node is set. However, we need to properly choose an energy source in order to comply with the above requirements. We will start by estimating the mean current consumption of the node, we will then determine the minimal battery size required. The solution given to the user is the one with the closest larger capacity.

Disconnection time slots: For this scenario, a minimal battery life per recharge is imposed. The node is already designed in terms of components and energy sources. Our goal is to determine the total amount of tolerated disconnection time. We will start by calculating the maximal mean current consumption allowed in order to respect the battery life constraint. We will then determine the disconnection period using a dichotomist search algorithm.

Design: In this case, a minimal battery life per recharge is to be respected. The daily routine of the patient is given. We have several commercial modules and energy sources to choose from. We would also need to properly configure these components. Our design tool will start by choosing, for each component, the most power efficient commercial module. It will then pick the smallest energy source that allows us to run the node with minimal performances. The last step, configuring these components, is done using an optimization algorithm.

Choosing an adequate optimization algorithm is crucial at this stage. Since the variables can be put in a discrete form, and the energy consumption increases with enhanced performances. We had to develop our own optimization algorithm. It is based on a brute-force search problem-solving technique. However, some features were added to make it more intelligent, thus reducing the amount of processing time. In fact, we defined an upper and lower limit of current consumption. The first is to respect the battery life imposed, while the second (-10%) aims to discard the solutions with the lowest performances. This algorithm will start by sorting the values taken by each variable from the one that produces the lowest consumption to the highest. It will then go through these values in order, keeping the ones that fit within the consumption limits. Whenever we reach the upper bound, the rest of the values are discarded without any calculation.

Another advantage is to take into account the user's preferences; for example, if a designer would rather work with a CPU frequency above 8 MHz, our algorithm will provide solutions that respect that constraint as well as the requirements of the application.

Application

Wireless body temperature sensor

In this section, we present an application. The sensor node we selected is a wireless body temperature sensor. Our choice was driven by the significance of body temperature in medical diagnosis. Table 2 matches each value with the corresponding sickness/symptoms.

The first step is to benchmark and look for components that provide low energy consumption coupled with good performances. We also took into account whether it is simple to assemble them in real life (supply voltage, size, etc.). We chose to work with Bluetooth Low Energy protocol because of its compatibility with laptops as well as smartphones. Table 3 matches each component with the commercial module we selected [8].

Estimating the current consumption of the node

Figure 6 displays the configuration of the problem. We consider that we are taking a measurement every two minutes for a month (30 days). Each one is a double digit precision value coded in ASCII (five bytes).

Figure 7 presents the current consumption recorded for the sensor, the analog to digital converter as well as the microprocessor. The state of the connection to the gateway has a major influence on the current consumption of the RF module. To understand this effect it requires a brief explanation of the Bluetooth Low Energy communication protocol.

In this technology, the connection takes place between a master/client (the gateway) and a slave/server (the node). Whenever it is not

Temperature	Symptoms
44	Death
43	Normally death, brain damage, cardio-respiratory collapse
41-42	Fainting, confusion, convulsion, Low/high blood pressure
38-40	Severe sweating, dehydration, weakness, vomiting, dizziness, fast heart rate
37	Normal temperature
36	Mild or moderate shivering
34-35	Intensive shivering, numbness, blue/gray skin, confusion, loss of movement in the finger
29-33	Confusion, sleepiness, slow heart rate, hallucinations
24-28	Breathing may stop, death

Table 2: Symptoms according to body temperature.

Component	Commercial module
Sensor	LM60
Processor	MS430FR5969
RF transceiver	Ble113 (Bluetooth Low) Energy)

Table 3: Components alongside the commercial modules chosen.

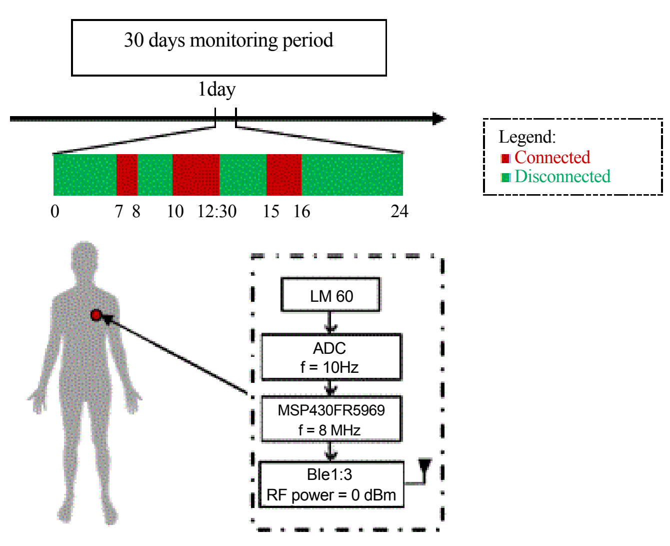

Figure 6: Configuration of the problem.

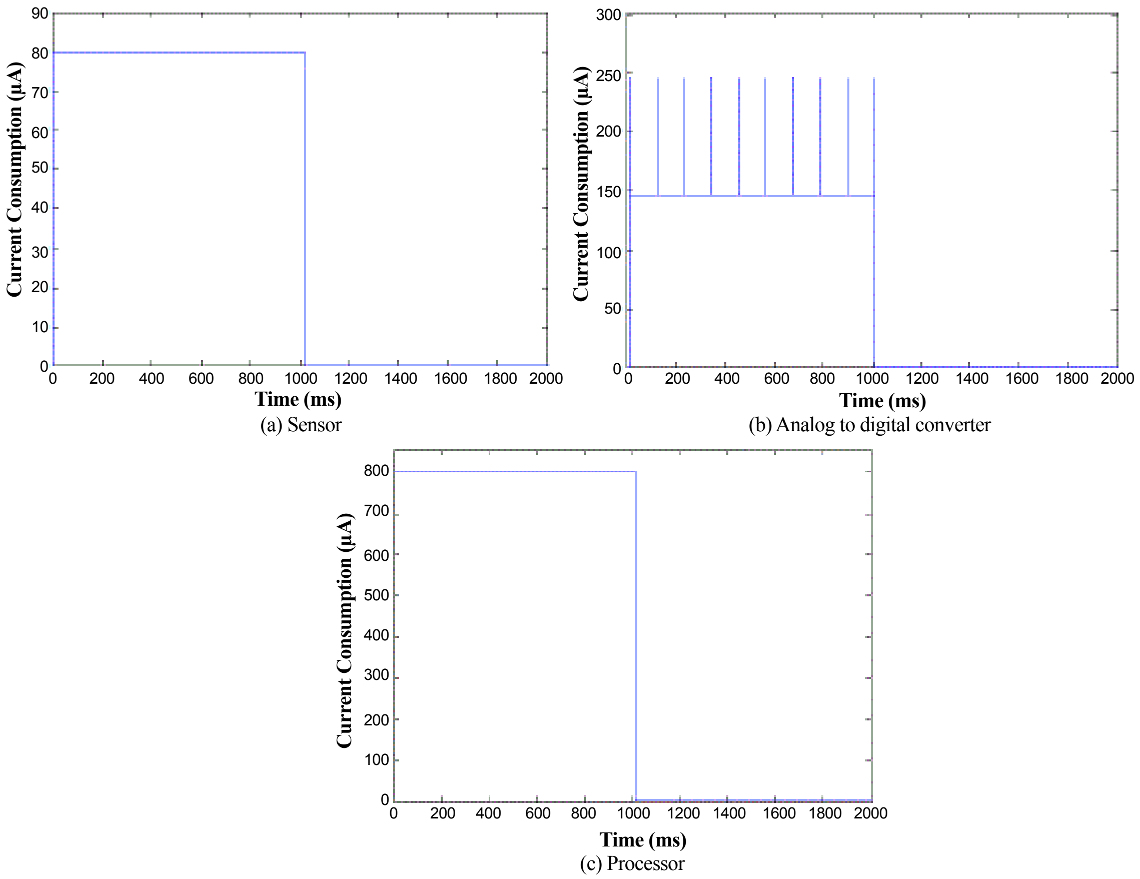

Figure 7: Current consumption registered in the sensor, the analog to digital converter and the processor.

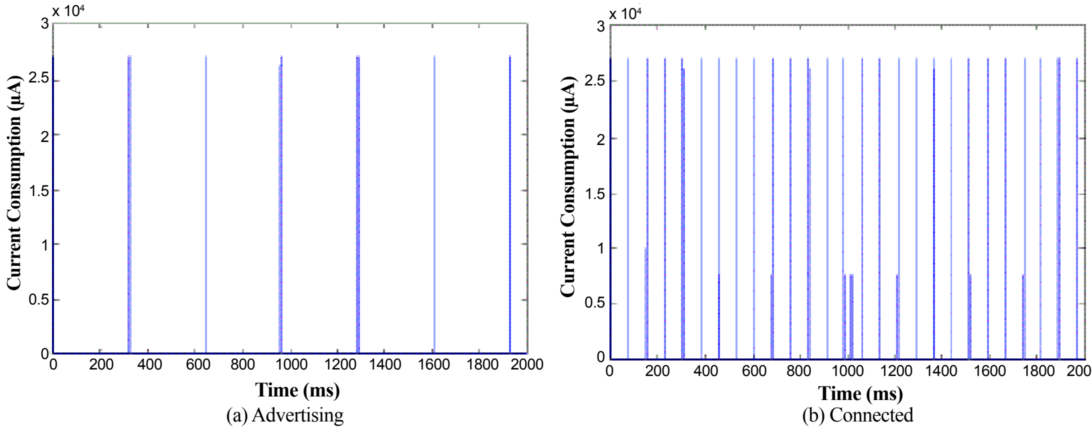

Figure 8: Current consumption registered in the RF module in advertising and connected modes.

connected, the server transmits data in advertising packets through three advertising channels. This transmission occurs in intervals of time called advertising events. After each sending of the advertising packets, the advertiser will be listening on the same channel for a while to check if there is a response coming from the gateway (i.e., the master). A connection starts whenever a master sends a connection request packet to the slave. The connection request packet can only be sent right after a successful reception of an advertising packet. It will contain connection related information such as the connection interval. Data transfer occurs within connection events. Such an event starts when the master sends a packet to the slave at the defined connection interval. The slave can respond 150 µs after its reception. It will send a packet that may contain data or not (connection maintenance packet) [15,16].

Figure 8 shows the current consumption of the ble113 module in both advertising and connected modes. We assume an advertising interval of 320 ms (default value in the ble113 module) and a connection interval of 75 ms (default value whenever connected to a bled112 dongle). The mean current consumption registered throughout the whole monitoring period is equal to 320.12 µA.

Guidance towards an energy aware design

Since the first three scenarios are quite straightforward, we will only be presenting an application for the Design solution.

Design: For this example, we assume we need to respect a 30 days minimal battery life as well as disconnection slots imposed by the daily routine of the patient. Moreover, we will be choosing between two microprocessors from TI's MSP430 series: the MSP430FR5969 and the MSP4305739. Table 4 summarizes their specifications as found in their respective datasheets. The MSP430FR5739 consumes less current in active mode than its counterpart but more in sleep mode. We will also be handpicking an energy source among the four batteries: CR2032 (230 mAh), CR2450 (620 mAh), 2xAAA (1150 mAh) and 2xAA (2100 mAh).

Our goal is to provide a user with the solution that includes the more power efficient components, the smallest possible battery and a configuration to efficiently use that energy source. This must be done according to the application requirements as well as the designer's preferences (Table 4). The proposed design included TI's MSP430FR5969 as the microprocessor and a CR2032 battery to power up the whole node. Several acceptable component's configurations are presented in Table 5. Figure 9 shows how our optimization algorithm brings us closer to the targeted consumption including each iteration.

Physical Implementation

Having associated our design tool with an application to a wireless temperature sensor, the next step is to physically implement the design. This will allow us to compare the results obtained through simulation with practical measurements.

This process is divided into three major parts: Prototyping the node, creating an application to collect data, and measuring the current consumption.

Prototyping a wireless sensor node

Adopting the Bluetooth Low Energy communication protocol required defining a Generic ATTribute profile (GATT). This profile mainly consisted of a "Body Temperature" service identified by a Universally Unique IDentifier (UUID). The service itself contains a "Body Temperature Measurement" characteristic. Whenever this characteristic's value is updated, the change will be indicated to the gateway. Indications are acknowledged.

Module	Current consumptions	
	Active mode	Sleep mode (LPM3)
MSP430FR5739	81.4 µA/MHz	6.3 µA
MSP430FR5969	100 µA/MHz	0.4 µA

Table 4: TI's MSP430 processors alongside their specifications.

Configuration of the node			
Measurements' rate	CPU frequency	Sampling frequency	RF power
Users preferences			
≥ One each 10 minutes	≥ 8 MHz	≥ 1 kHz	0 dBm
Proposed configurations			
One each 5 minutes	8 MHz	10 kHz	0 dBm
One each 5 minutes	8 MHz	1 kHz	0 dBm
One each 10 minutes	16 MHz	10 kHz	0 dBm
One each 10 minutes	16 MHz	1 kHz	0 dBm

Table 5: The proposed configurations with respect to the designer's preferences.

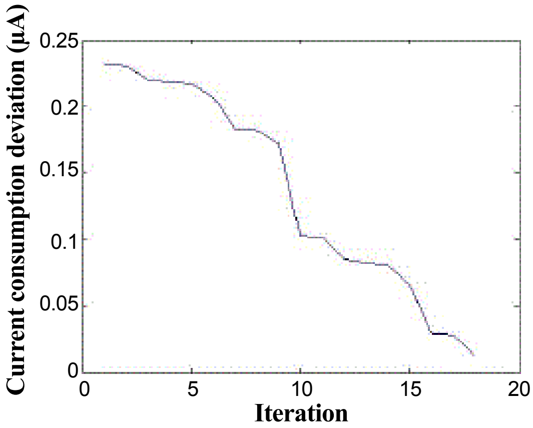

Figure 9: Evolvement of the current consumption deviation with each iteration.

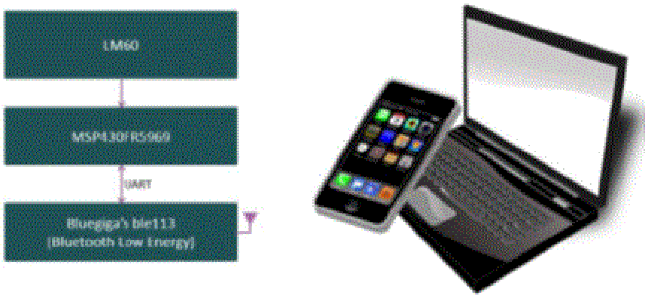

Figure 10: Block diagram of the temperature sensor node.

Figure 10 shows the block diagram of the node we prototyped. Its operation is summarized by the following:

The measurement's rate is defined using the microprocessor's Real Time Clock;

The processor will wake up periodically according to this rate;

It will power up the sensor through one of its pins for 1 second;

During that time an analog to digital conversion takes place in the processor's analog to digital converter with a 10 Hz sampling frequency;

The temperature's initial value is retrieved in the processor;

It's then passed to the Bluetooth Low Energy module;

The Bluetooth Low Energy module updates the "Body Temperature Measurement" characteristic's value.

In order to achieve minimal current consumption, modules, timers and peripherals were put to sleep when not in use. Moreover, current leakage through pins had to be avoided.

Collecting data

At this stage, we had to create a PC application in order to collect the incoming data. Such software must be able to scan for all the available connections. However it must only connect to the one advertising the desired service. The next step is to look for certain characteristics in order to read, write, activate notifications/indications, etc. We chose Python as a programming language in order to benefit from multiple libraries allowing us to plot data, write it into files, access date and time, etc. The application we created will scan for Bluetooth Low Energy modules looking for the "Body Temperature" service. It will then connect to the corresponding ble113 module and activate indications for the "Body Temperature Measurement" Characteristic. Once this is done, it will start receiving temperature related information. Real time plots as well as a data log are updated with each value.

Current measurements

Since both Texas Instruments and Silicon Labs prioritize power consumption, the developments kits were made in a way to make it easy for the user to determine the current consumption of the module. That's why we have decided to work with these kits for current measurements.

Sensor (LM60), processor and analog to digital converter (MSP430FR5969): These components are powered from the MSP430FR5969. Thus, measuring the latter's consumption will take into account theirs as well. Measurements were done using TI's Energy Trace [17] and an ampere meter [18]. The first outputs the power consumption's diagram as well as the mean value for current and voltage. The second will allow us to measure the current consumption in sleep mode without any influence from the debugger.

Radiofrequency module (Ble113): The first step would be to compare current consumption waveforms to the ones measured with a scope. This measurement, later referred to as V_0, is done over a 3 ohm resistor using an instrumental amplifier with a gain of 10. It will result in an inverted signal and the current consumption I can be calculated based on the basic following equation [19]:

$I=(3.3-V_0)/30$

This will allow us to measure the current registered during transmission and reception. It will also enable us to measure the connection and the advertising intervals. The next step is to accurately measure the module's current consumption in sleep mode using an ampere meter [19].

Results

In this section, we compare results obtained through simulation with practical measurements in order to discuss relevance of our design tool.

Sensor (LM60), processor and analog to digital converter (MSP430FR5969)

For experimental reasons, we are sending temperature data each 5 seconds. For this setup a mean current consumption of 153.7 μA was measured. Figure 11 matches the power measured in real life with the one simulated. Both have the same one second active period. However the consumption in active mode is much higher in simulation than in real life.

Since the results obtained do not match those of the simulation, additional measurements were made. The deviation recorded earlier was due to the current registered in active mode: this current depends on the cache hit ratio, i.e., the size and structure of the program (Table 6) [20].

We then took all these measurements and tuned our design tool. The mean current consumption obtained by simulation was 146.25 μA. Since Energy Trace technology requires a connected debugger, we had to take into account its consumption (4 μA) in idle mode and deduce it from the measured mean current consumption. The results are shown in Table 7.

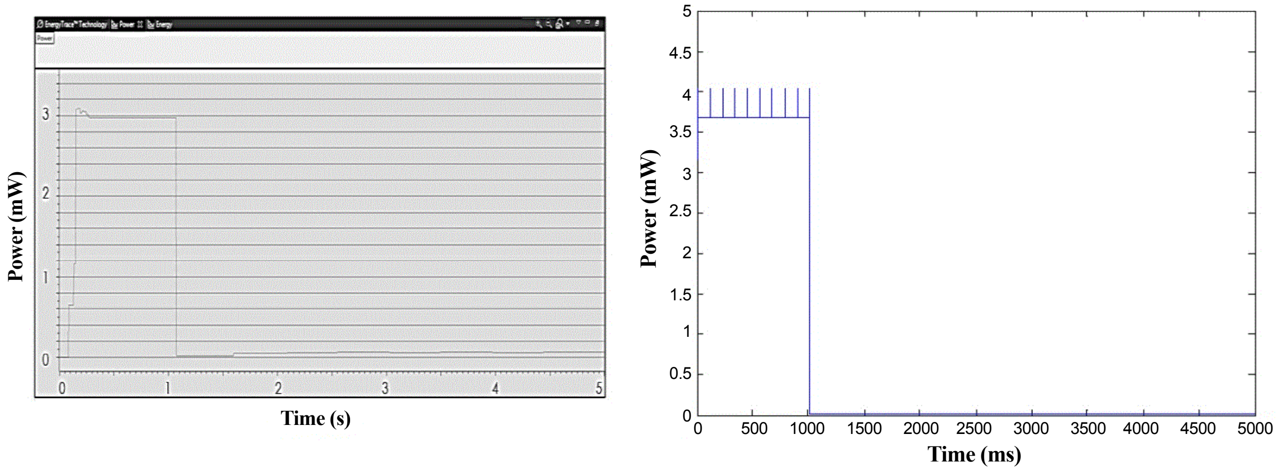

Figure 11: Power measurements obtained through simulation compared to the one measured with energy trace.

Cache hit ratio (%)	Current consumption in active mode (μA)
00	2510
50	1440
66	1070
75	890
100	420

Table 6: Current consumption registered in the MSP430FR5969 according to the cache hit ratio for an 8 MHz CPU frequency.

Current consumption (µA)	Simulation	Real life measurements	Relative error
Mean value	146.25	149.7	-3.5 %

Table 7: The mean current consumption measured compared to the one obtained through simulation for the sensor-ADC-processor sequence.

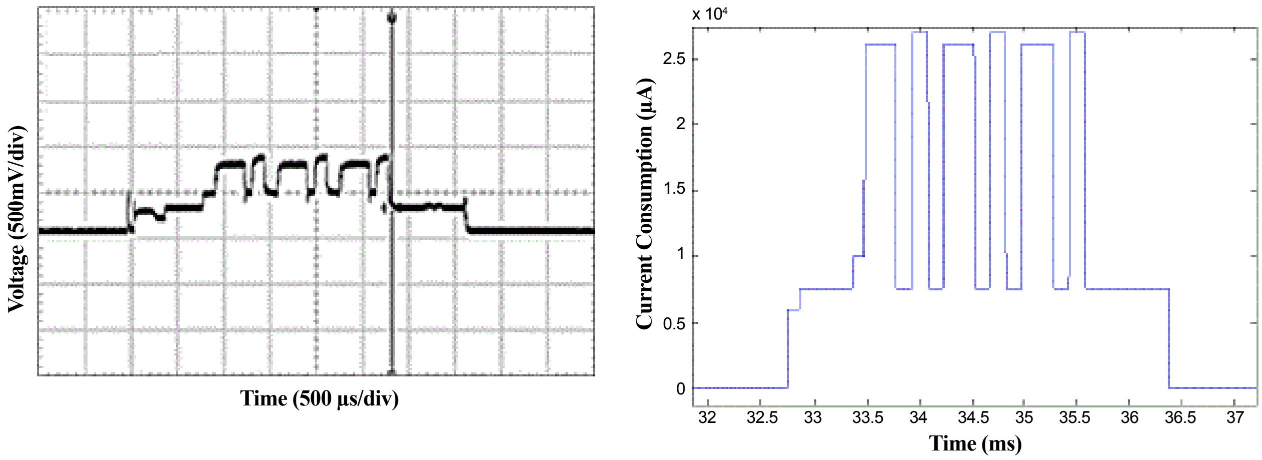

Figure 12: Current consumption measured throughout a single advertising event matched with the one simulated.

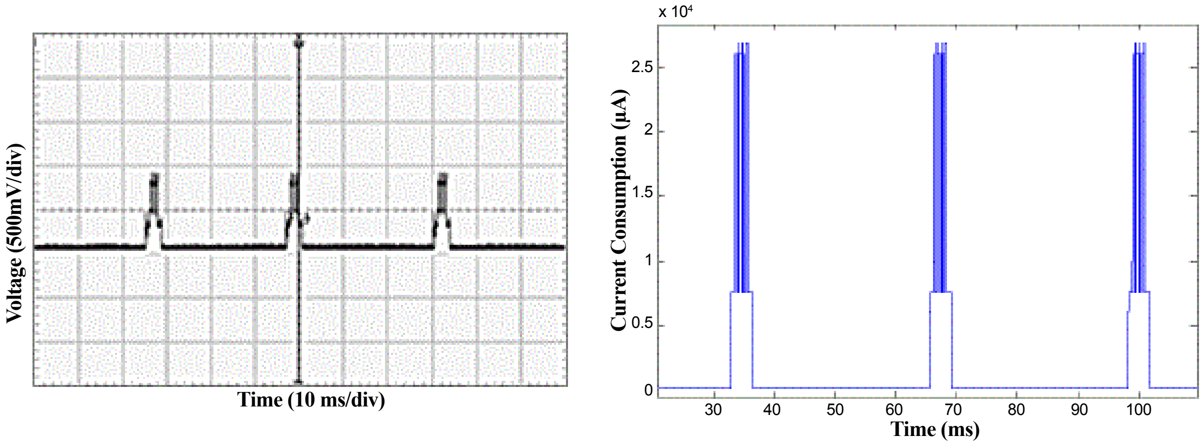

Figure 13: Current consumption measured during several advertising events matched with the one simulated.

Radiofrequency module (Ble113)

We first matched the waveforms measured with the scope with the ones produced by simulation. Note that we reduced the advertising interval (32 ms). This will result in more accurate measurements although it will significantly increase the mean current consumption. Figure 12 is related to the advertising mode. It zooms in on a single advertising event allowing us to measure the currents in transmission as well as reception. It is clear that both waveforms are similar in terms of transitions and time spent in each state. Figure 13 also displays more than 1 advertising event allowing us to measure the advertising interval and compare it with the value we set. Figures 14 and Figure 15 provide the same information in connected state. Table 8 matches each measurement taken with its value as provided in the module's datasheet and used in our design tool [21].

Errors recorded over transmission, reception and active currents are slightly higher. This is due to the current consumption of the measurement setup (the 3 ohm resistor and the instrumental amplifier) [19]. This extra consumption is not a constant and depends on the current consumed by the ble113 module. Moreover, it cannot be accurately measured since the time spent in each state is relatively small. The measurement setup was disconnected when measuring the current in sleep mode (PM2).

Concerning the advertising interval, configuring it in the ble113 was done by defining a minimal and a maximal value which explains the minimal error observed. On the other hand, the connection interval was set by default as a single value in the Bluetooth Low Energy dongle. That is why the error was 0%.

Furthermore, the data obtained from simulation is similar to the one measured in the physical implementation of the temperature sensor node in terms of the consumption waveforms as well as the mean values. Thus, our design tool allows the user to predict the consumption of a node prior to its implementation with small error.

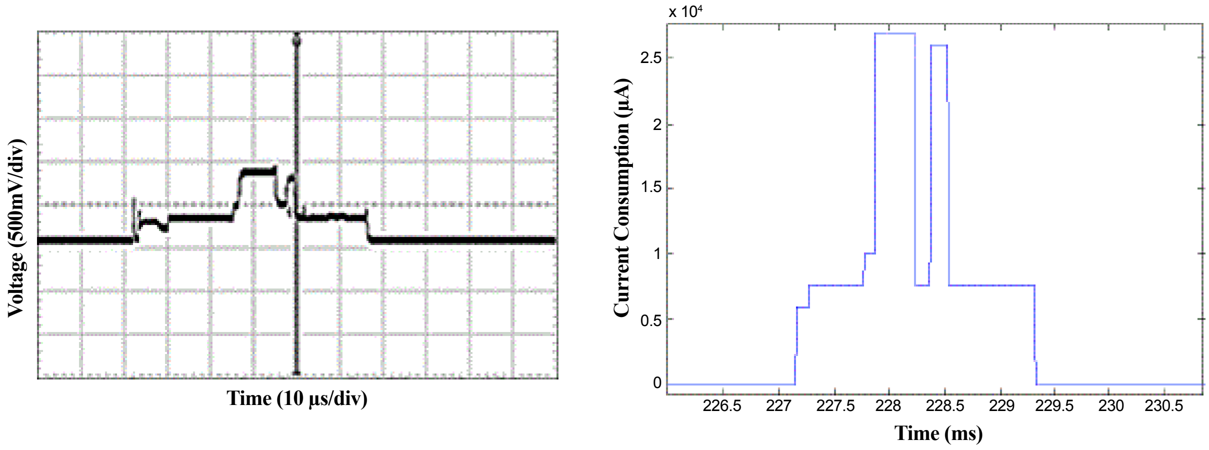

Figure 14: Current consumption measured during a single connection event matched with the one simulated.

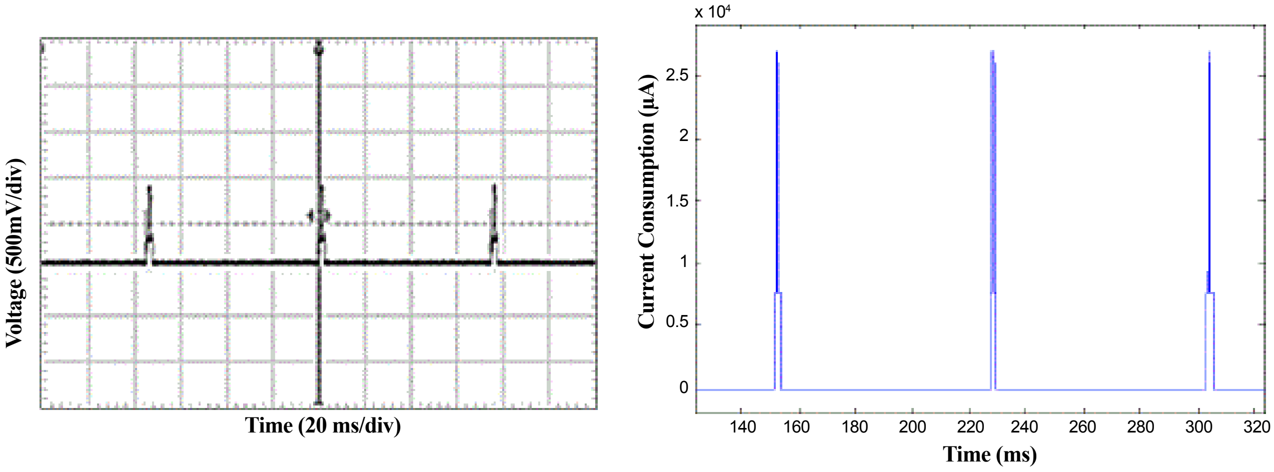

Figure 15: Current consumption measured during several connection events matched with the one simulated.

Data measured	Real life measurements	Datasheet	Relative error
Transmission current (mA)	28.3	26.1	+7.77%
Reception current (mA)	29.3	27	+7.78%
Active current (mA)	8	7.6	+5.3%
Sleep current–PM2 (µA)	0.93	0.9	+3.22%
Advertising interval (ms)	31.8	32	-0.62%
Connection interval (ms)	75	75	0%

Table 8: Real life measurements compared to the values found in the datasheet of the ble113 module.

Conclusion

This paper presented a Matlab based design tool for wireless sensor nodes used in remote patient monitoring. It is power oriented and enables early design explorations. It has many built-in solutions to guide the user during an energy aware design process. It can also be easily used for different setups. The deviation between simulation and experimental measurements recorded over the mean value is around 5%.

In order to guarantee a patient's wellbeing, it is vital not to lose any information. Storing data over a memory chip will ensure that and make the system more reliable. Thus it is mandatory to model memory access operations, guide the user to the data backup strategy that best suits his application and study its effect over the consumption of the node.

In addition, most medical conditions require monitoring simultaneously more than one vital sign. This requires several nodes operating within a network. However, the energy consumption as well as the quality of service depends on the number of sensors and the amount of data to transmit by each.

Furthermore, it would be interesting to include other applications such as industrial applications, civil structure monitoring, security and surveillance, smart buildings, etc.

Finally, creating a user interface would further reduce the time and effort required to design.

Acknowledgement

Rammouz R, gratefully acknowledges the support of The Azm and Saade Association - Lebanon for funding his work. A part of this work is funded by the Lebanese University research program.

A part of this work is funded by the Lebanese University Research Program.

References

1. Yuce MR, Khan J (2011) Wireless body area networks: technology, implementation, and applications. CRC Press.
2. Hanson MA, Powell Jr HC, Barth AT, Ringgenberg K, et al. (2009) Body area sensor networks: Challenges and opportunities in Computer, pp: 58-65.
3. Movassaghi S, Abolhasan M, Lipman J, Smith D, Jamalipour A (2014) Wireless body area networks: A survey in IEEE Communications Surveys & Tutorials. IEEE 16: 1658-1686.
4. Noury N, Fleury A, Nocua R, Poujaud J, Gehin C, Dittmar A (2009) Sensors for medical monitoring. Sensors, algorithms and networks in IRBM 30: 93-103.
5. Henderson TR, Roy S, Floyd S, Riley GF (2006) ns-3 project goals. Proceeding from the 2006 workshop on ns-2: the IP network simulator 13.
6. Tiwari V, Malik S, Wolfe A (1994) Power analysis of embedded software: a first step towards software power minimization, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 2: 437-445.
7. Chen G, Branch J, Pflug M, Zhu L, Szymanski B (2005) SENSE: a wireless sensor network simulator. In Advances in pervasive computing and networking Springer US, pp: 249-267.
8. Galos M, Navarro D, Mieyeville F, O'Connor I (2012) A cycle-accurate transaction-level modelled energy simulation approach for heterogeneous Wireless Sensor Networks. In New Circuits and Systems Conference (NEWCAS) 2012 IEEE 10th International, pp: 209-212.
9. Molina JM (2010)Energy profiling of networked embedded systems, Ph.D thesis, Vienna University of Technology, Vienna.
10. Terrasson G (2008) Contribution à la conception d'un émetteur-récepteur pour microcapteurs autonomes, Ph.D Dissertation, Université Sciences et Technologies-Bordeaux I, Bordeaux.
11. Rammouz R, Labrak L, Abouchi N, Constantin J, Zaatar Y, et al. (2015) A generic Simulink based model of a wireless sensor node: Application to a medical healthcare system. Proceedings International Conference on Advances in Biomedical Engineering (ICABME), pp: 154-157.
12. Casilari E, Cano-García JM, Campos-Garrido G (2010) Modeling of current consumption in 802.15. 4/ZigBee sensor motes. Sensors 10: 5443-5468.
13. Negri L, Beutel J, Dyer M (2006) The power consumption of Bluetooth scatternets. In Proc. IEEE Consumer Communications and Networking Conference (CCNC), pp: 519-523.
14. Gomez C, Oller J, Paradells J (2012) Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology in Sensors 12: 11734-11753.
15. Dementyev A, Hodges S, Taylor S, Smith J (2013) Power consumption analysis of Bluetooth Low Energy, Zigbee and ANT sensor nodes in a cyclic sleep scenario. Proceedings IEEE International Wireless Symposium (IWS), pp: 1- 4.
16. Liu J, Chen C, Ma Y, Xu Y (2013) Energy analysis of device discovery for Bluetooth Low Energy networks. Proceedings IEEE 78th Vehicular Technology Conference (VTC Fall), pp: 1-5.
17. Texas Instruments (2014) MSP430TM Advanced Power Optimisations: ULP AdvisorTM and EnergyTraceTM Technology. Application Report.
18. Texas Instruments (2014) MSP430FR5969 LaunchPadTM Development Kit (MSP-EXP430FR5969) User’s Guide.
19. Silicon Labs (2013) Ble113 Development Kit 2.0 datasheet.
20. Texas Instruments (2012) MSP430FR59xx Mixed Signa Microcontrollers datasheet.
21. Silicon Labs (2015) Bluegiga Bluetooth Smart Software v.1.4 API Documentation.