Handbook of Research on Mobility and Computing:

Evolving Technologies and Ubiquitous Impacts

Maria Manuela Cruz-Cunha
Polytechnic Institute of Cávado e Ave, Portugal

Fernando Moreira
Universidade Portucalense, Portugal

Volume I

INFORMATION SCIENCE REFERENCE
Hershey · New York

Senior Editorial Director: Kristin Klinger
Director of Book Publications: Julia Mosemann
Editorial Director: Lindsay Johnston
Acquisitions Editor: Erika Carter
Development Editor: Michael Killian
Typesetters: Michael Brehm, Casey Conapitski, Keith Glazewski, Milan Vrarich Jr. & Deanna Zombro
Production Coordinator: Jamie Snavely
Cover Design: Nick Newcomer

Published in the United States of America by
Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com

Library of Congress Cataloging-in-Publication Data

Handbook of research on mobility and computing : evolving technologies and ubiquitous impacts / Maria Manuela Cruz-Cunha and Fernando Moreira, editors.
p. cm.
Includes bibliographical references and index.
ISBN 978-1-60960-042-6 (hbk.) -- ISBN 978-1-60960-043-3 (ebook) 1. Mobile computing. 2. Wireless communication systems. I. Cruz-Cunha, Maria Manuela, 1964- II. Moreira, Fernando, 1969 Aug. 16-
QA76.59.H35 2011
004.165--dc22
2010036723

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Chapter 62

A Low Cost Wireless Sensors Network with Low-Complexity and Fast-Prototyping

João Paulo Carmo
University of Minho, Portugal

José Higino Correia
University of Minho, Portugal

ABSTRACT

This chapter presents a low cost/fast prototyping wireless sensors network that was designed for a huge range of applications and making use of low cost commercial of the shelf components. Such applications includes industrial measurements, biomedical, domestic monitoring, remote sensing, among others. The concept of the wireless sensor network is presented and simultaneously, hot topics and their implementation are discussed. Such topics are valuable tools and can't be discarded when a wireless sensors network is planed. By the contrary, such tools must be taken in account to make the communications between the nodes and the base station the best possible reliable. The architecture, protocols and the reasons that were behind the selection of the components are also discussed. The chapter also presents performance metrics that are related to with the physical characteristics of sensors and with the radio specificity.

Microcontrollers with a RISC architecture are used by the network nodes to control the communication and the data acquisition and operate in the 433 MHz ISM band with ASK modulation. Also, in order to improve the communication and to minimize the loss of data, it is predicted to put the wireless nodes to handle line and source coding schemes.

This chapter cover the following topics: • the focus and application of the wireless sensor network; • the implications of the radio system; • the test bed implementation of the proposed low cost wireless sensors networks; • the wireless link power budget, coding and data recovering; • performance metrics of the wireless sensors networks; • cost analysis versus other technologies (wired and emerging wireless).

DOI: 10.4018/978-1-60960-042-6.ch062

INTRODUCTION

Wireless communication microsystems with high density of nodes and simple protocol are emerging for low-data-rate distributed sensor network applications such as those in home automation and industrial control (Choi *et al.*, 2003). It is available a huge range of solutions to implement wireless sensors networks (WSN). A few companies (Crossbow 2009; Dust 2009; Sensicast 2009) are offering products such as radios (motes) and sensor interfaces. The motes are battery-powered devices that run specific software. In addition to running the software networking stack, each mote can be easily customized and programmed, since it runs open-source operating systems which provides low-level event and task management. Mote Processor/Radio module families working at 2.4 GHz ISM band that support IEEE802.15.4 and ZigBee are available. However, to implement wireless buses for certain applications, it is required compact and miniaturized solutions. Also, the inclusion of chip-sizes antennas in the RF microsystem can be a crucial factor, as it is the case presented in (Enz *et al*., 2005) to target applications in wearables. However and despite their easiness, these solutions can revealed very expensive when it is desired to deploy a industrial network prototype. Thus, low-cost and ready-to-deploy solutions are more attractive for the Portuguese's small-and-medium industries (PMEs), as it is the case of restaurants and snack-bars, where it is mandatory to keep temperature logs in frizzing cameras with a periodicity less than an hour. If this regulation is not implemented and respected, the ASAE (*Autoridade de Segurança Alimentar e Económica*) organism acts in conformity and penalties starting from simple monetary dues to the close of the facilities are consequences to keep the activities working out of the law. Data acquisition systems require automated and efficient processes to do the records and logging. A wired infrastructure can be one possible solution. However, this can be a problem especially in older facilities, where holes must be made in the walls to pass the cables. The installation of wireless infrastructure is another way to install a communication connection. A wireless infrastructure allows the installation of multi-hop networks without doing severe changes in the facilities. Also, this kind of solution has the advantage to increase the number of network nodes with high flexibility. Behind that, other nodes with other type of functions can be installed. Also, since the prototyped solutions don't follows the mass production and thus the low-cost per unity, a new and prototypable solution must be found to meet these small-volume applications.

The wireless sensors network presented in this chapter meets a wide range of small-volume applications with a low-cost and in a ready to use fashion.

IMPLICATIONS OF THE RADIO-FREQUENCY SYSTEM

Normally, in the majority of the wireless sensors network applications, the total power consumption of a wireless node has a low or negligible contribution due to the electronics of control and processing, when compared with those from the radio-frequency (RF) system. The simple matter of fact that the available technologies present increased low-power features, it is not synonymous of a total power consumption relief. This is justified due to the fact the RF transceiver to be the bloc with the highest power consumption (Enz *et al*., 2005). The usage under low periods of time or low-duty cycles is a key to save power in wireless nodes. As depicted in Figure 1, the duty-cycle is defined by the ratio *duty-cycle* = T_u/T_f, where T_u [s] is the working time of the network for a total life time, T_f [s] and must be low. This paradigm is useful and it corresponds to what happens in a real wireless sensor network, where the nodes work in a peaked based transmission (Mateu *et al*., 2007).

Next, use of clock frequency is a sensitive topic because if the transceiver is not transmitting neither receiving, it is advisable to use of the smallest clock frequency in the local signal processing to save even more power. An additional strategy to save even more power, is to put the wireless node to sleep when the processing is finished (Bicelli *et al*., 2005).

Another way to optimize and reduce the power consumption is the exploration of two new key-factors (Cho *et al*., 2004). These factors are the start-up and the transmission times. The first one is the time between an order of enable is given to the electronics and the instant these same electronics starts effectively to work. The second is the time that lasts the complete data transmission. The reduction of these times helps to reduce the power consumption in the transmitter side. Normally, nodes for low-power applications has low temporal cycles of working (*duty-cicles*), as well as low packet lengths, which are very short, thus, the start-up time can have a significant impact in the whole power supply. In the context formerly presented, the transmitter must send the data in the lowest period of time (high baud-rates), while simultaneously must present the lowest start-up time. To better understand this concept, the Figure 2 shows the starting time, t_a [s], versus the transmission time, t_{TX} [s]. In this scenario, the *duty-cicle* is\

$$duty - cycle = T_u / T_f = \sum_k (t_a + t_{TX}) / T_f = \sum_k (t_a + N_b / r_b) / T_f \quad .$$

Assuming by simplicity that the bit number, N_b, is fixed and unchanged, the start-up time, t_a, has a significant impact in applications with high bit-rates, r_b [bps]. In this situation, the start-up time, t_a, becomes predominant, when compared with the transmission time, t_{TX}, in the numerator of the *duty-cicle*.

An additional technique to save power, is to process data before to be transmitted. Low volumes of data, requires less time in the transmission, e.g., it implies low power consumptions (Akyildiz *et al*., 2002). Furthermore, the loss of data or receptions with errors must be avoided, in order to don't have unnecessary wastes of power (Mackensen *et al*., 2005). Moreover, the nodes must be able to select the lowest but suitable power transmission, in order to save power to. As illustrated in Figure 3, a receiver strength indicator (RSSI) is of major interest to achieve this goal. Basically, a RSSI is an envelope detector followed by a logarithmic amplifier (Analog Devices, 2009). Known the transmitted power and once the received power is obtained. The next step is to select the power of transmission.

Unfortunately and contrary to the transmitter, the number of available options to the receiver

Figure 1. Illustration of the low-period usage concept

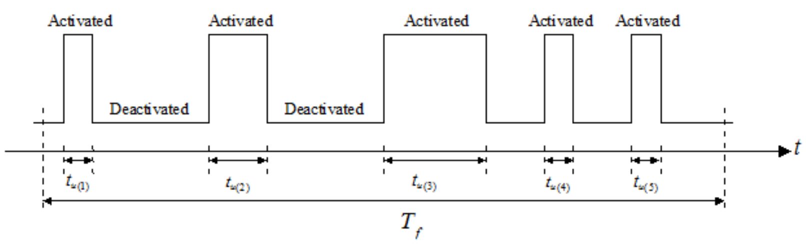
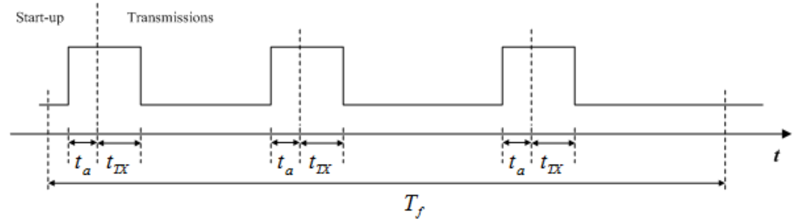
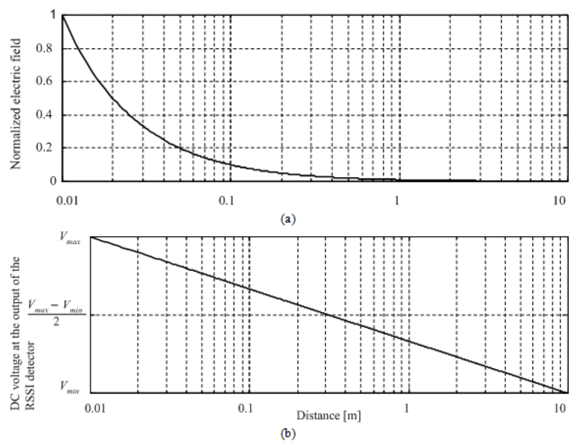


Figure 2. Illustration of the effect of the start-up time in the whole power consumption of a RF transmitter



are very limited, because this one can't know exactly when a data transmission is targeted to it, thus, the receiver must always be activated and receiving data (Mackensen *et al*., 2005). The only solution, is to use the RSSI circuit detect the presence of a carrier with a significant power and use this event to wake-up the network node. Even the used modulation can be a limiting factor, due to the power consumption. A remind must be made in order to say that compared with a simple narrow amplitude modulation (AM), the use of a direct sequence spread spectrum (DSSS) technique available in the IEEE 802.15.4 has the advantage to make the data transmission more reliable, with the cost of an increasing in the power consumption (Callaway *et al*., 2002; Gutierrez *et al*., 2001). To

Figure 3. (a) Electric field strength indication as function of the separation between the emitter and the receiver, d [m]; and (b) a the respective DC voltage [V] at the output of RSSI detector



finish, in wireless communications, the antenna is one of the most critical subsystem, thus, in order to not compromise the desired miniaturization, the antenna must be small enough to comply with size constraints of the microsystems. The investigation of new frequency bands (Celik *et al*., 2008) and new geometries (Mendes *et al*., 2008) will make possible to have smaller antennas to integrate in wireless microsystems (Touati *et al*., 2006). This makes the chose of the most suitable frequency, one of the more decisive aspects in the design of RF transceivers. Normally, the desired range, baud-rate and power consumptions are key-aspects in the design to take in account, when the frequency of operation is to be selected. At a start-up point, the range limits the maximum usable frequency, because the loss suffered by radiowaves in the free-space increases with the distance. However, to keep or even increase the useful life of the batteries, such a variation in the transmitted power is not possible to do. Moreover, in the case of applications requiring high baud-rates, the transmitted bandwidth must also be high, in order to support these applications. However, the frequency can't be arbitrarily increased, because this have implications in the power consumptions, e.g., at high frequencies, the transistors must switch faster, thus the energy dissipation will be higher.

IMPLEMENTATION OF THE WIRELESS SENSORS NETWORK

System Architecture

The proposed wireless sensors network has nodes constituted by a Microchip's PIC16F628 microcontroller and by a set composed by a sensor read-out connected to an analog-to-digital converter (the TI's TLC0820 ADC) of eight bits, and digital circuits to control the read-outs (where the TI's CD74HC165 parallel-to-serial converter is a key-component). This microcontroller is responsible to provide the basic services for communication and control. The core services also allow the extension of the node's functionalities with additional services. The block diagram of the network nodes is showed in Figure 4. These network nodes has the formerly cited sensors read-out, the RF interface (the Radiometrix's Bim433) and an optional RS-232 interface (the Maxim-IC's MAX233) to transfer data towards an external computer, a PDA or a mobile phone. More than one sensors can be connected to the wireless node with the sacrifice of the sampling frequency, $f_S^{(N)}$ [Hz], given by $f_S^{(N)}=f_S^{(1)}/N_{sensors}$, where $f_S^{(1)}$ [Hz] is the maximum sampling frequency for wireless nodes with only one sensor and $N_{sensors}$ is the number of sensors. As seen in Figure 4, the number of sensors per wireless node is done defining a modular architecture based on parallel-to-serial circuits (the TI's CD74HC165) that multiplex the acquired signals in the digital domain.

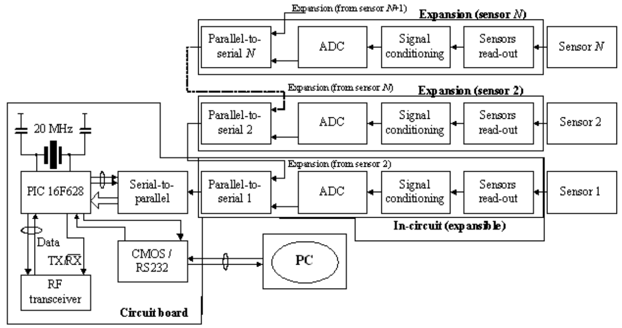
The prototype uses a commercial RF transceiver, which operates at 433 MHz. The microcontroller PIC16F628 was select, due to its frequency clock of 20 MHz, which corresponds instructions with an execution speed of 0.2 μs. Using this clock, and the maximum baud-rate of 40 kbps imposed by the RF transceiver, a total of five hundred (#500) instructions are executed for each transmitted bit. However, and as will be discussed further, the implemented line code reduces the effective baud-rate to half, e.g., doubling the processing time for each transmitted bit.

Frame Formatting

As depicted in Figure 5, two types of frames were defined: the general use and the command frames.

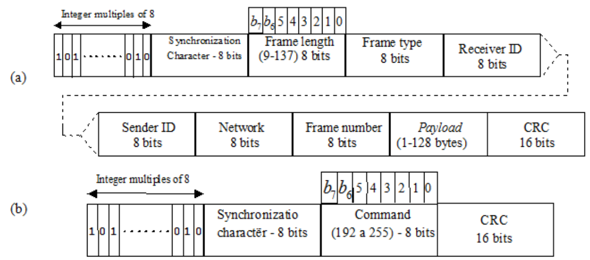
The general frames has two purposes, one is to carry information in the payload field between the nodes and the base-station, in a coordinator fashion. The second function is to send commands from the base-station to the network nodes. The command frames are used by the base-station to send commands toward the network nodes that

Figure 4. The block diagram of a node prototype for wireless sensors network



were already identified by the base-station, where the need to identify it, is not needed. These frames sends commands that are quickly identified, such as confirmations of good (ACK - Acknowledgements) or bad reception (NACK - Nacknowledgements) of previously received data. In the first frames, the payload length is variable. In the case of this frame be used to send commands, the field **Frame type** is 01h (00 00 00 01b), its length is minimum and is of only nine bytes. The default case is when the frame carries data, e.g., the value in the **Type field** is 00h (00 00 00 00b). In the future, additional types can be defined, for values in the **Type field** of 02h (00 00 00 10b) or higher. These frames, allows to identify the destiny (the receiver), to numbering the network and to check with the help of the CRC field, the existence of transmission errors. This is also allowed in command frames.

Figure 5. Fields in (a) the general use and in b) the control frames



Line Coding

This is perhaps the most important issue in the WSN. Very long sequences of ones or zeros can result in a data imbalance, which can cause the loss of carrier and bad symbol synchronisation. To have a good data balance, e.g., one level transition for a set of two consecutive data bits, a sequence of two symbol bits are transmitted at twice the effective baud-rate (the data-rate). The symbol bits sequences '10' and '01' are transmitted, when the information bit '1' or '0' is to be send. Moreover, this scheme also helps to synchronise the clock of the receiver with the clock of the transmitter (Carlson, 1986).

As illustrated in Figure 6, before a node send the byte $b_7b_6b_5b_4b_3b_2b_1b_0$, a program call must be made, to divide that byte in two parts, and to create two new (separated) bytes $b_7b_7b_6b_6b_5b_5b_4b_4$ and $b_3b_3b_2b_2b_1b_1b_0b_0$. If the older byte belong to the header, then the exclusive or (XOR) is executed in the two new bytes, using the mask "01 10 01 10b". However, if the older byte don't belong to the header, then the XOR is made with the mask "01 01 01 01b". Independently the result of the XORs, the two resulted bytes are transmitted at the twice the data-rate of the information contained in the frame. If the user choices to not code the frames, then the same program is also called, but the mask is always "00 00 00 00b". In this case, a data balancing will not ensured. Compared with the coded case, and in order to have a real double data-rate, the software must double the processing rate.

The appendix A shows a portion of assembly code responsible to do the line coding.

Synchronisation of Frames

To a correct reception of frames, the receiver must evaluate with accuracy the start of the frames. As depicted in Figure 7, this is done using a window, which is no more than a FIFO with a capacity of 16 bits, which is filled with the symbol bits as they arriving. This window starts to fell the presence of the header, and as soon as the synchronisation

Figure 6. Manchester masks applied (a) in coded and (b) in uncoded frames

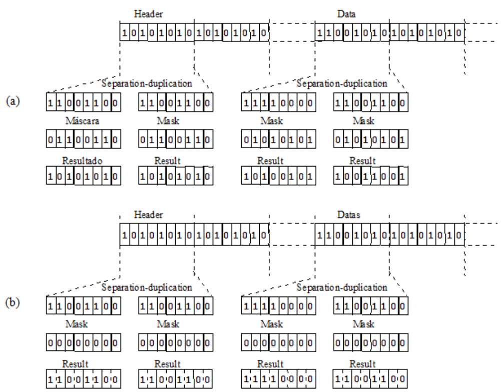
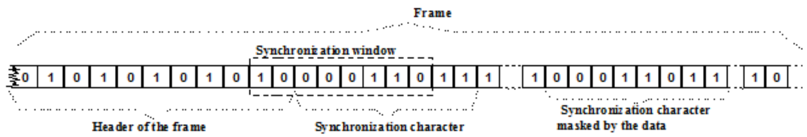


Figure 7. Window to detect the synchronisation character 1Bh (00 01 10 11b)



character (FAW) is fully received and fully fills the FIFO, then the reception of the frame and the data in the payload fields will start to happen. Figures 7 and 8, illustrates this process taking the synchronisation character 1Bh (00 01 10 11b) as an example.

Error Controlling

The data transmission is not immune to errors in the channel. Thus, it was defined a error control field with a length of sixteen bits, in the footer of both types of frames, e.g. the CRC (cyclic redundancy check) field. The CRC is correlated with the transmitted data. After receiving the entire frame, the receiver make the calculation of CRC of that frame and then compares this value with the CRC contained in the footer of the frame. If both CRCs are equal, the receiver assumes that the data were received without errors. In the opposite case (inequality of the CRCs) the data has errors.

The CRC is generated according the polynomial (Microchip, 2000) $p(x)=a_{16}x^{16}+a_{15}x^{15}+a_{14}x^{14}+\ldots+a_2x^2+a_1x+a_0$. The values a_k are zeros or ones, and imposes the existence of each of the feedback connections illustrated in Figure 9.

Figure 8. Acquired received base-band signal, where it is possible to observe the header and the synchronisation character 1Bh (00 01 11 10b), which is Manchester coded, "01 01 01 10 - 10 01 10 10b"

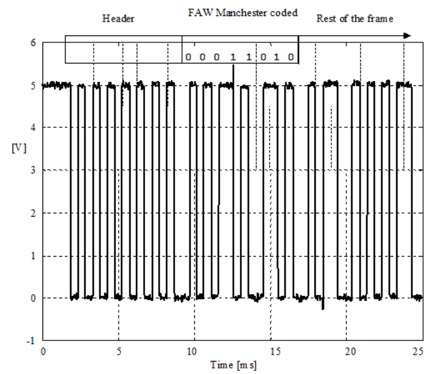
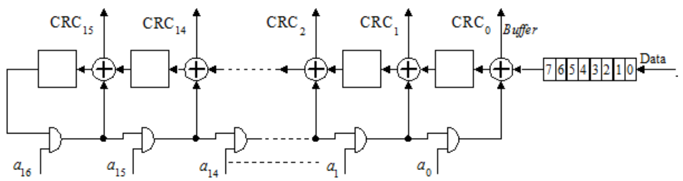


Figure 9. Generation of the CRC



The CRC generation is very simple, and it is based on a calculator (CALCproc) procedure, which is called the number of the bytes to be used. The content of the shift-register (SR) of Figure 9 is cleaned and after an execution of the CALCproc, its value remains in the SR, in order to be available to the next byte to be processed. The CALCproc has an eight bits buffer to store and to make eight shifts during each call. The values CRC_{15} to CRC_0 give the temporary CRC number to be transmitted. This number also remains in the SR, until the last byte be fully processed, which is the CRC number to be encapsulated in the frame. After a complete frame construction, the SR is cleaned again and will be ready to the next CRC generation.

Some portions of the CRC generation source code can be observed in the Appendix B.

Hardware Specificities of Hardware Implementation

The Figure 10 shows how the buffering is done by the PIC16F628 microcontroller (Microchip, 2009). The PIC16F628 is based on the Harvard architecture and has two types of memory: the flash memory (or program memory) and the data memory. The flash memory uses 2 kByte to store the program, while the data memory is used to store environment variables and other type of data. Contrary to the program memory, the data memory has a limited capacity and it provides two (selectable) banks, each one with 128 bytes. The data acquisition from sensors is temporarily stored in these two banks. A total of 128 bytes is provided by both memory banks (64 in each bank) and a trade-off exists between the sampling frequency, the number of nodes, and the minimum transmission bit-rate.

Figure 10. Buffering illustration inside the PIC16F628. (a) Flash memory containing the stored program, and two banks of 128 bytes that contains the user's data: (b) BANK 0 (memory positions 00H to 0FH) and BANK 1 (the remaining 0FH positions)

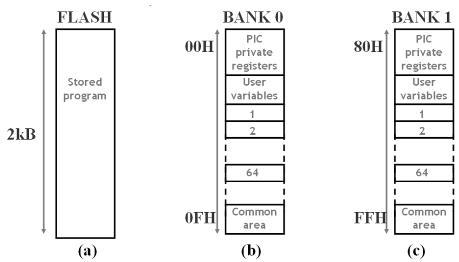
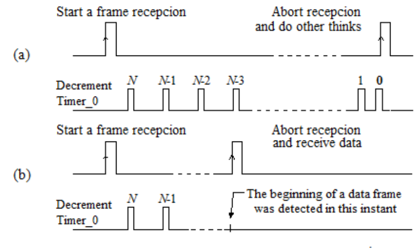


Figure 11. How to use Timer 0, to prevent dead-locks



In order to avoid deadlocks, it was implemented timeout mechanisms. The timeout makes a node to avoid the situation to be eternally waiting to receive a frame, which will never arrive. Moreover, the synchronization of the receiver's clock with the transmitter was implemented, in order to avoid the loss of frames, due to bad timing references.

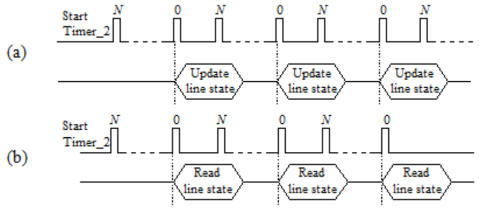
The timeout detection was make with the use of the PIC16F628's Timer 0 (Microchip, 2009), which is set to a given value before a receiving operation to take place. A periodically decrement is made to this timer, while the start of a receiving frame is not detected. If the content of the Timer 0 reaches the null value, then a timeout event is declared by the node, and the receiving operation is aborted. A clock with a frequency of 20 MHz, allows fine variations of 13.1072 ms and coarse variations of its multiples. Figure 11(a) shows how Timer 0 can prevent a potential dead-lock, whereas the Figure 11(b) shows how a good reception is handled by this Timer.

As show in Figure 12, Timer 2 was used to synchronize receiver with the transmitter (Microchip 2009). This timer is always initiated with the same value (PV), and it is putted to run continuously without stops. Then, every time a overflow is experimented, it will auto-initiate with the previously value, PV. The electrical state of the transmitting line is updated (e.g., a new bit is transmitted), whenever an overflow occurs. In this situation the flags are cleaned and it will start all again for the next bit. The new bit is putted in the line only with a new overflow. In the receiver's case, the process is identical. The crystals used in to provide the clock to the microcontrollers presents deviations from the nominal frequency of oscillation. This is not a problem for short frames, where the error integration can be neglected. In order to avoid the loss of data, for a clock with a tolerance of $\pm p$ [ppm], the number of bits, N_b, in the frame must be less than $(1\pm p)/p$.

Effect of Errors in the Loss of Data

The less severe effect is when one or more bits in the payload are toggled. The most severe effect is when at least one bit in the address of the destiny (the receiver ID field) is toggled. In this situation, the receiver wrongly discard a frame with data.

Figure 12. Management of the (a) transmission and (b) reception of frames



However, bit changes in other five important fields (synchronization character, frame length, frame type, network ID and frame number) also implies (total or partial) loss of data. For a channel with an error probability, BEP, the probability, P_{loss}, to occur a loss of frame:

$$P_{lost} = \sum_{k=1}^{\text{Total frame length}} C(48,k).\mathrm{BEP}^k \approx C(48,1).\mathrm{BEP} = 48.\mathrm{BEP} \qquad (1)$$

where $C(n,k)$ is the number of k-combinations from a set with n elements. In a data frame, n is the number of sensitive bits, e.g., n=6×8 bits. These are the bits susceptible to generate loss of frames, when errors are present in the channel. Six is the number of important fields in a data frame (receiver ID, synchronization character, frame length, frame type, network ID and frame number), where a single error will result in a loss of frame.

ACQUISITION VERSUS TRANSMITTING TIMES

The acquisition time and the comparison with the processing time is of extreme importance to know the loss of samples. Some assumptions are taken in advance to simplify the analysis: first, it is considered a network with N nodes located d_k [m] (where k=1...N_{nodes}) from the base-station and ready to transmit frames with a length of $N_{oct,k}$ data bytes after few data acquisitions. Also, the summing of the processing times in the transmitter, t_{proc_TX} [s], is constant and equal for all nodes, and the same applies with the processing time of receivers, t_{proc_RX} [s]. If the base-station has enough memory storage capacity, then the baud-rate must be at least:

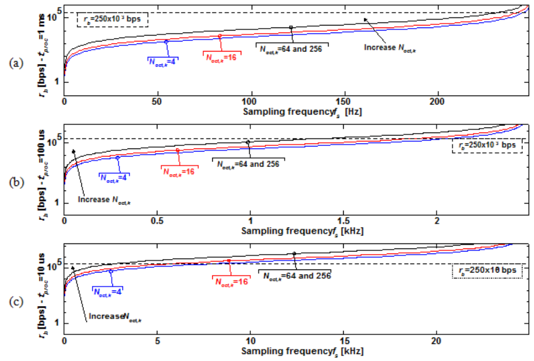
$$r_b > \frac{13+\max_k(N_{oct,k})+N_{ctl}+2N_{header}}{\frac{1}{f_s}-\frac{2d_k}{c}-2(t_{proc\_TX}+t_{proc\_RX})} \qquad (2)$$

where N_{header} is the number of bits in the header.

Three typical scenarios are considered, where the processing times, t_{proc_TX} and t_{proc_RX}, are both equal to 1 ms, 0.1 ms and 0.01 ms. Also, the wireless modules are close each others (with a distance d_k of 10 meters), and the number of bytes in the payload, $N_{oct,k}$={4, 16, 64, 256} bytes (which corresponds to 1, 4, 16 and 64 samples of 2 analog channels of 2 bytes each).

The Figure 13 shows the minimum baud-rate to avoid the loss of frames for several sampling frequencies and simultaneous number of bytes in the payload. The Figure 14 shows the minimum

Figure 13. For three processing times, t_{proc_TX}=t_{proc_RX}, a) 1 ms, b) 0.1 ms and c) 0.01 ms: the minimum baud-rate, r_b [bps], versus the analog sampling frequency, f_S [Hz], considering data frames whose payload's length, $N_{oct,k}$, are 4, 16, 64 and 256 bytes (these two last situations have similar behaviors, and overlap)



baud-rate to avoid the loss of frames for several sampling frequencies and simultaneous number of bytes in the payload. From Figure 14, it is evident that as high is the sampling frequency, the high must be the baud-rate in order to be delivered more data during the same time. Moreover, as the processing time t_{proc}=t_{proc_RX}=t_{proc_TX} increases, the high must be the baud-rate, in order to compensate the waste of time during the processing. Another conclusion is that as high is the number of bytes in the payload, the high must be the baud-rate, in order to not lose again, a frame with data. Another important aspect is the negligible effect of

Figure 14. For three processing times, t_{proc_TX}=t_{proc_RX}, a) 1 ms, b) 0.1 ms and c) 0.01 ms: the minimum baud-rate, r_b [bps], versus the analog sampling frequency, f_S [Hz], considering data frames where, $N_{oct,k}$={ 4, 16, 64 and 256} bytes (these two last situations have similar behaviors, and overlap)

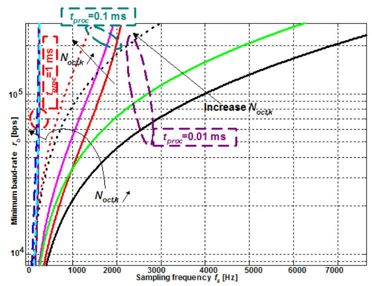
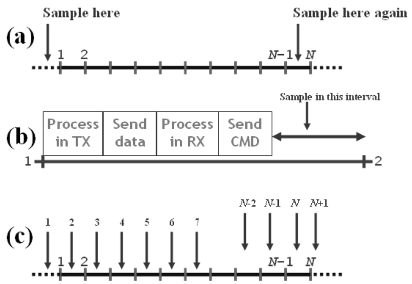


Figure 15. Sequences of operations: (a) a special case, (b) timming diagram with the operations, and (c) the generic case



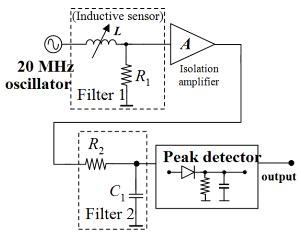
the spacing d_k in the baud-rate, which practically means that t_{proc} must be smaller than $1/(4f_s)$.

The sequence of operations is illustrated in Figure 15(b) and it can be seen that the case depicted in the projection (a) is the one that allows the lowest baud-rate, r_b, whose expression is a special case of the previous equation. Also, this special case allows low analog sampling frequencies [Hz] (or high multiplexed analog signals sampled at high frequencies). The more general equation (2) applies to the generic case illustrated in Figure 15(c), where both the baud-rate and the sampling frequencies (or the number of multiplexed signals) are present.

$$r_b > f_s\left(N_{nodes} \times \left[9 + \max_k\left(N_{oct,k}\right) + \max_k\left(N_{control,k}\right) + 2N_{header}\right]\right) \times \times 1 / \left[1 - f_s \times N_{nodes} \times \left[\frac{2\max_k\left(d_k\right)}{c} - t_{proc\_TX} - t_{proc\_TX}\right]\right] \quad (3)$$

The measurement of the breath rate, a transducer with a variable inductance that indirectly measures changes in the thoracic diameter is an example of application for the proposed wireless

Figure 16. The block diagram of a signal conditioning used in the measure of the breath-rate



sensors network. The device is located in a position around the body at the level of maximum respiratory expansion. At maximum inspiration the belt is stretched almost to maximum extension, making the inductance minimum. As depicted in Figure 16, a frequency of 20 MHz can be used in this circuit. The variations in the inductance, changes the attenuation for the 20 MHz signal, in the first *RL* low-pass of first order filter (at Filter 1). Thus, the attenuation increases when the thorax perimeter decreases. This filtered signal is further amplified, before a second low-pass filtering (at Filter 2), to eliminate noise and spurs generated in the 20 MHz oscillator. Then, a peak detector gets the amplitude of the 20 MHz processed signal, which is:

$$V_{peak} = \underbrace{\frac{V_{20}.A}{\sqrt{1 + w^2C^2R_2^2}}}_{\text{Fixed term}} \times \underbrace{\frac{1}{\sqrt{1 + w^2L^2 / R_1^2}}}_{\text{Variable term (with } L)} \quad (4)$$

where $w=2\pi\times20\times10^6$ [rad.s^{-1}] is the angular frequency, V_{20} [V] is the amplitude of the 20 MHz sinusoidal signal and A is the gain of an auxiliary amplifier. The voltage at the output of the peak detector is amplified to cover the entire input range of the analog-to-digital converter (ADC). Also, the former amplifier provides isolation between the two filtering stages, which helps to avoid load-matching problems, when these stages are connected. The signal at the output of the peak detector follows the low-frequency respiration signal and it is converted to the digital domain by the TLC0820 ADC

CONCLUSION

This paper presents a fast-prototyping with low-cost wireless sensors network nodes, which were developed with low-cost off-the-shelf commercial components. Such a wireless nodes can be used in applications that ranges from the simple industrial to a more sophisticated biomedical application, as it is the measurement of the breath rate.

REFERENCES

Akyildiz, I. F. (2002). Wireless sensor networks: a survey. *Computer Networks*, *38*(1), 393–422. doi:10.1016/S1389-1286(01)00302-4

Analog Devices, (2009). AD8309 datasheet.

Callaway, E. (2002). Home networking with IEEE 802.15.4; a developing standard for low-rate wireless personal area networks. *IEEE Communications Magazine*, *45*(8), 2–9.

Celik, N. (2008). Implementation and experimental validation of a smart antenna system operating at 60 GHz band. *IEEE Transactions on Antennas and Propagation*, *56*(9), 2790–2800. doi:10.1109/TAP.2008.928785

Cho, S. (2004). A 6.5-GHz energy-efficient BFSK modulator for wireless sensors applications. *IEEE Journal of Solid-state Circuits*, *39*(5), 731–739. doi:10.1109/JSSC.2004.826314

Choi, P. (2003). An experimental coin-sized radio for extremely low-power WPAN (IEEE 802.15.4) applications at 2.4 GHz. *IEEE Journal of Solid-state Circuits*, *38*(12), 2258–2268. doi:10.1109/JSSC.2003.819083

Crossbow, (2009). Micaz module, Wireless measurement systems. Crossbow Inc., Retrieved March 09, 2010 from www.xbow.com.

Dust, (2009). Dust Networks Inc., Retrieved March 09, 2010 from www.dust-inc.com.

Enz, C., et al. (2005, November). *Ultra low-power radio design for wireless sensor networks*. Paper presented at the IEEE Int. Workshop on Radio-Frequency Integration Technology, Singapore. Bicelli, S., *et al*. (2005, September). *Implementation of an energy efficient wireless smart sensor*. Paper presented at the XIX Eurosensors, Barcelona, Spain.

Gutierrez, J. (2001). IEEE 802.15.4: developing standards for low-power low-cost wireless personal area networks. *IEEE Network*, *15*(5), 2–9. doi:10.1109/65.953229

Mackensen, E., et al. (2005, September). *Enhancing the lifetime of autonomous microsystems in wireless sensor actuator networks (WSANs)*. Paper presented at the XIX Eurosensors, Barcelona, Spain.

Mateu, L., et al. (2007, October). Paper presented at the 2007 International Conference on Sensor Technologies and Applications, Valencia, Spain, October.

Mendes, P. M. (2006). Integrated chip-size antennas for wireless microsystems: Fabrication and design considerations. *Sensors and Actuators. A, Physical*, *123-124*(1), 217–222. doi:10.1016/j.sna.2005.07.016

Microchip, (2000). CRC generation and checking. Application note 730.

Microchip, (2009). PIC16F628 data sheet, 16-pin Flash 8-bit microcontroller.

Sensicast, (2009). Sensicast Systems, Retrieved March 09, 2010 from www.sensicast.com.

Touati, F., et al. (2003, September). Paper presented at the ESSCIRC, Estoril, Portugal. Carlson, B., (1986). *Communication systems - 2nd edition*, McGraw-Hill.

KEY TERMS AND DEFINITIONS

Wireless Sensors Network: A network comprising wireless nodes, whose principal function is to acquire physical measures and send them wirelessly towards a base station. Also, these nodes can work as relays to other nodes or directly to the base station, and it must possess the ability to temporarily store its acquired data and the data to be forwarded;

PIC: A family of microcontrollers from Microchip Inc;

Radio-Frequency (RF): Frequencies used to transmit data across the air;

Radio: Electronic system used to transmit and receive RF signals;

RF Transceiver: The same as radio;

Frame: Set composed of bits (unities of binary information) arranged in a logic sequence;

Error: A change (toggle) in the bit value due to channel impairments (noise, interferences, multipath, among others).

APPENDIX A – LINE CODING SOURCE CODE

```
LINE_CODING
        ; Variables (in the PIC16F628 Bank_0):
        ; InByte:   Byte to be coded (8 uncoded bits)
        ; OutByte1: Coded bits - bits 7-4 from InByte (8 coded bits)
        ; OutByte2: Coded bits - bits 3-0 from InBtye (8 coded bits)
Coding_Nible_LSB
        MOVLW         B'00001111';         Mask 4 LSBs from InByte to code
them
        ANDWF         InByte,W
        ADDWF         PCL,F;             Jump to one of these positions:
        GOTO         LSB_0;                     InByte = xx xx 00 00
        GOTO         LSB_1;                     InByte = xx xx 00 01
...................
        GOTO         LSB_15;                    InByte = xx xx 11 11
        GOTO         LSB_Error;                 InByte = xx xx ?? ?? (ER-
ROR??????)
LSB_0
        MOVLW         B'01010101';         InByte = xx xx 00 00 --> OutByte1
= 01 01 01 01
        MOVWF         OutByte1;
        GOTO         Coding_Nible_MSB
...................
LSB_15
        MOVLW         B'10101010';         InByte = xx xx 11 11 --> OutByte1
= 10 10 10 10
        MOVWF         OutByte1;
        GOTO         Coding_Nible_MSB
LSB_Error
        MOVWF         B'00000000';         OutByte1 INVALID (some kind of er-
ror has happened)
        MOVWF         OutByte1;        ;
Coding_Nible_MSB --------> DO THE SAME with the 4 MSBs of InByte and construct
OutByte2
```

APPENDIX B – CRC GENERATION SOURCE CODE

```
CRC_CALCULATOR
;Input data:        Auxiliary ---> Byte to treat (shifting 8 times)
;                 CRC_LOW  is pré-inicialized (00 00 00 00  in the first time)
;                 CRC_HIGH is pré-inicialized (00 00 00 00  in the first time)
;                 Px_HIGH  ---> Generator polinómio (MSByte)
;                 Px_LOW   ---> Generator polinomy (LSByte)
;Output data:
;                 CRC_LOW  Updated (not coded 8 bits)
;                 CRC_HIGH Updated (not coded 8 bits)
;
        MOVF        Auxiliary; Load CRC_BUFF with the byte to treat
        MOVWF        CRC_BUFF
        MOVLW        0x08; NBit = 8 pass to the WREG (ALLWAYS!!)
; Shifting of  CRC_HIGH <-- CRC_LOW <-- CRC_BUFF to the left
CRC_Shifting
        ADDLW        0x00; To clean the Carry flag (Don't affect W register)
        RLF        CRC_HIGH,F; Shift CRC_HIGH to the left
        ADDLW        0x00; To clean the Carry flag (Don't affect W register)
CRC_CALCULATOR - Continuation
        BTFSC        CRC_LOW,7; Bit 7 from CRC_LOW is 0?
        BSF        CRC_HIGH,0; NO! Set Bit 0 from CRC_HIGH, before rotate CRC_
LOW.
        RLF        CRC_LOW,F; Shift CRC_LOW to the left
        ADDLW        0x00; Clean Carry
        BTFSC        CRC_BUFF,7; Bit 7 from CRC_BUFF é 0?
        BSF        CRC_LOW,0; NO! Set bit 0 from CRC_LOW, before rotate CRC_
BUFF.
        RLF        CRC_BUFF,F; Shift CRC_BUFF to the left
        ; Can I Apply the CRC Polinomy?
        BTFSC        CRC_HIGH,7; MSB do CRC_HIGH is 1?
        GOTO        Apply_Polinomy_CRC; NO
        GOTO        Decrement_NBit; NO
Apply_Polinomy_CRC
        MOVWF        Auxiliary; Save WREG (with the NBit) in the Auxiliary
        MOVF        Px_HIGH,W; Apply MSByte (Px_HIGH) of the Polinomy in the
CRC_HIGH
        XORWF        CRC_HIGH,F; CRC_HIGH = CRC_HIGH (Xor) Px_HIGH
        MOVF        Px_LOW,W; Apply LSByte (Px_LOW) of Polinomy in CRC_LOW
        XORWF        CRC_LOW,F; CRC_LOW = CRC_LOW (Xor) Px_LOW
        MOVF        Auxiliary,W; Put WREG with the previously saved value
```



```
Decrement_NBit
        MOVWF        Auxiliary
        DECFSZ        Auxiliary,F; Decrement NBit. NBit = 0?
        GOTO        Cycle_CRC; NO
        GOTO        Out_Calculator_CRC; YES ---> Go out with CRC Updated
Cicle_CRC
        MOVF        Auxiliary,W
        GOTO        CRC_Shifting
Out_Calculator_CRC
RETURN; Go out from CRC_CALCULATOR_CRC
```