Remote supervision system for aquaculture platforms

Manuel Sousa e Silva
Faculty of Engineering
University of Porto
manelsousaesilva@gmail.com

Nuno A. Cruz
Faculty of Engineering and
INESCTEC
University of Porto
nacruz@fe.up.pt

Fernando P. Lima
Faculty of Sciences and
CIBIO/InBIO
University of Porto
fplima@gmail.com

***Abstract*—Aquaculture processes usually take place in remote and harsh environments, and are highly dependent on uncontrollable and unpredictable variables, therefore its monitoring and supervision can be a key factor in this activity. Taking that into account, this paper proposes a solution for a Remote Supervision System for Aquaculture Platforms, that contemplates a modular, reconfigurable and expandable sensor network based on the I2C protocol, which is composed by two different types of sensor nodes. The *main sensor node*, which serves as the sensor network coordinator and as a gateway, and the *tiny sensor nodes*, that are responsible for simple data collection tasks.**

***Index Terms*—Aquaculture gateway, embedded systems, remote monitoring, sensor network.**

I. Introduction

Aquaculture processes typically take a considerable amount of time, usually months, and a single unwanted event can ruin a whole harvest. This loss of several months work often happens without having means or information to understand its main cause. The Monitoring and Logging of those environmental conditions is essential to understand behaviors and dynamics related with the development of the species. That information could allow the producers to prevent external factors impacting the production.

In the long term, the analysis of this data could generate knowledge that would increase and improve the know-how of the producers, which would be able to enhance their processes during the development and growth of the species. Therefore, these systems can be a key factor in the future of this industry.

The logging of these variables can also be very relevant for ecological purposes. Having specific data concerning different environmental parameters about an individual and its surrounding environment enables the study and correlation of all that information, which might be the basis for some evolution in the understanding of ecosystems dynamics.

There are a considerable number of commercial solutions available to perform these tasks. However they still have various constraints, the solutions are not customized for each experiment or environment; usually it is mandatory to use them using the supplier's software; and these solutions usually represent high costs.

II. State of the Art

Researchers started to use simple stand alone devices, the **data loggers**, to sample environmental data. By the year of 2000, Whiteman *et al.* [1] stated that, at that time, the advances in electronics miniaturization had finally allowed the commercial development of inexpensive temperature sensor and data loggers solutions. Simple and battery powered devices were already capable of operating autonomously for long periods of time, even doing some internal processing of the measured data.

Heidemann and Govidan [2] defined **embedded sensor network** as a network of embedded computers that interact with its surrounding environment. This idea entailed three essential concepts: nodes, motes and sensors. Nodes and motes, at different levels, are responsible for the data acquisition, processing, and for the communication with the network. Sensors represent the lower level in this architecture, as they are responsible for measuring the desired variable.

This new trend of uploading all the data collected to an external database lead to a new concept, known as Wireless Sensor Networks (WSN). In this paradigm, all the sensing and data logging devices, which may be deployed over a large area, are connected in a wireless network capable of transmitting all the data the sensors gathered to an external system. [3]

A **Gateway** is a key concept in these communications networks, and it is essential in order to understand the sensor networks paradigm, as it can be seen as a protocol converter. A Gateway is a network node capable of interfacing with a different network, which uses another communications protocol, but also can serve as network coordinator controlling others devices and gathering the information collected by the sensor network in order to transmit it.

A. Related Works

As already referred, the commercial solutions for this kind of purpose are only satisfactory for a minority of scenarios with simple requirements. Therefore the scientific community is constantly developing devices in order to answer to the opportunities not covered by the solutions available in the market, or to propose different, simpler, and cheaper solutions.

There are several projects published in scientific literature, that are not directly associated with aquaculture, such as

978-1-5090-1537-5/16/$31.00 ©2016 IEEE

devices oriented for telemedicine or oceanography. Its end purpose might not be the same, but the technical solutions are of high interest as an inspiration to the work under development.

Blank *et al.* [4] presented a prototype of a device projected for telemedicine purposes. The described module is responsible for the data collection and communications management of a network of wearable sensors. Essentially, it is a gateway to enable the transmission of sensor data to a web server.

This system main operation is based in a microcontroller that interfaces with several devices and sensors. It it has three different ways to communicate with the server: Ethernet, Wi-Fi and GSM. However, for applications where the expected tasks to be performed by the system are more demanding in terms of computational processing, it is possible to attach an embedded Linux computer (Beaglebone). In this situation the Beaglebone assumes the system operation and the microcontroller serves as an interface with all the peripherals.

Another interesting architecture is proposed by Chianese *et al.* in [5], where the authors present a smart multisensor framework for a sensor network, which sensor nodes are based in Beaglebone Black boards.

In this work the sensor nodes are all based in identical devices, but they can assume one of four possible roles in the network: *Stand-Alone, Client, Monitor or Server*, depending only on the software loaded to the board. The role each sensor node assumes in the network depends on the specific application where this framework is employed.

It is important to refer that this is only possible, as the Beaglebone Black has great interfacing capabilities, that is fundamental for its operation as *Client* collecting sensor data. It also has a considerable computational power that can process the software responsible for the operation of the network, when the sensor node is working as *Monitor or Server*.

Manikandan *et al.* [6] work focuses on the application of the I2C protocol in a sensor network. They state that this protocol is mostly used within a PCB and between ICs, but they present a single master and multi-slave architecture with several sensors externally connected to a microcontroller. The authors conclude that the system can be extended, and use an I2C bus as the basis for a large cabled sensor network.

III. System Overview

The development of this system is the result of a partnership between FEUP (Faculty of Engineering), and CIBIO/InBIO, a research Institute also associated with the University of Porto.

CIBIO/InBIO biologists understood the significant importance that electronics can have in the gathering of information for their studies and hence started developing some systems using low cost electronics products and open source software [7]. CIBIO/InBIO has been involved in a project with the Spanish authorities in Galicia, and has a clear interest in using the sensing method they developed [8], in order to collect data regarding mussel behaviour, more specifically its heartbeat using a simple IR sensor.

This previous experience and the know-how resultant from their previous works was crucial in the requirements assessment for the system to be developed in this project.

The system to be developed faced several challenges, as it should be able to acquire underwater data from a considerable number of sensors (10 to 200), manage and store the acquired data, and transmit the sensor data to a predefined destination. All these tasks were expected to be executed autonomously by the system.

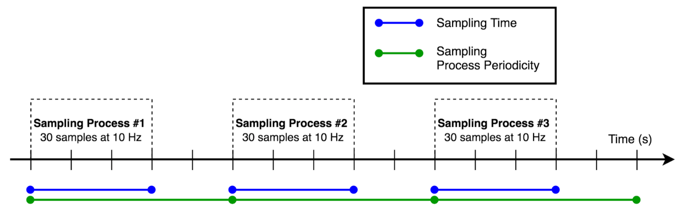

Figure 1. Possible sampling setup example.

Finally, as shown in figure 1, when defining the system sampling orders for each sensor, there are three parameters that must be definable: number of samples, sampling frequency at which the sample must be collected, and the sampling process periodicity, i. e. how often each sampling process should occur.

IV. Proposed Solution

Taking into consideration the system requirements, the proposed solution contemplates the development of an embedded sensor network composed by several sensor nodes.

In order to fully explain the concept developed, figure 2 presents a possible topology for the system. The proposed architecture contemplates a reconfigurable sensor network, where extra sensor nodes can be added to the bus. This network is composed by two different types of sensor nodes:

- 1 **Main Sensor Node** that serves as the sensor network coordinator, but also functions as a the network gateway.
- 1 to 200 **Tiny Sensor Nodes** that are responsible for simple sampling tasks.

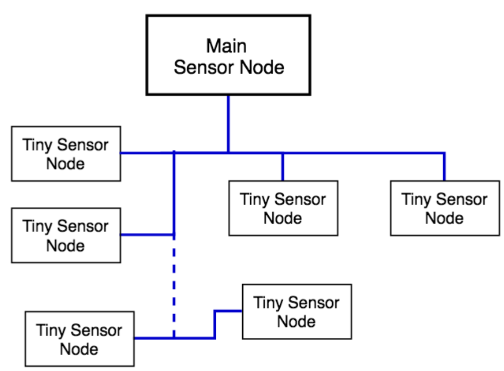

Figure 2. Possible topology for the proposed sensor network.

For the sensor network, an I2C bus was used. This protocol has been chosen since it does not present constraints in the design of the network topology and allows an easy network reconfiguration. It also allows a reduction of the wiring complexity and consequently the costs with all the cabling. The main sensor node can be set as the network master and the tiny sensor nodes as slaves since it is possible to attribute an address to each one. Using this communications method it is possible to connect up to 128 nodes in a single bus, with this number being limited by the bus capacitance, which can be solved using I2C repeaters for data transmission in long lines [9].

In Figure 3 it is possible to see the architecture of the most complex sensor node, which serves as a gateway for the whole network, as it is responsible for coordinating the whole sensor network, being capable of gathering the data collected by all the sensor nodes, store that information, and whenever a network is available, transmit the sensor data to a previously defined destination.

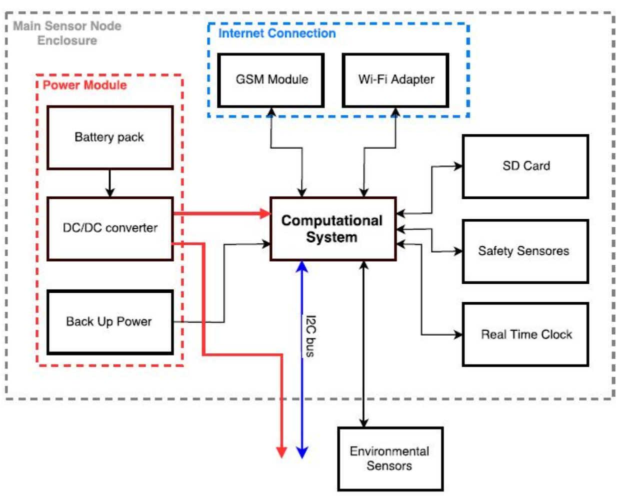

Figure 3. Main Sensor Node Architecture

Since this system is responsible for coordinating the sensor network and gathering a considerable amount of data, it requires some computational power. Therefore, the core of this main sensor node is based in a computational system, that has I/O capabilities, in order to manage a network constituted by less complex sensor nodes, and to read some local sensors to acquire data regarding its surrounding environment.

Another important task to be ensured is the Internet Connection, that can be achieved either by a Wi-Fi adapter, if any network is available, or by a GSM Module. That can prove to be an interesting option as this kind of system are typically deployed in relatively remote locations where a usable Wi-Fi network is unavailable but the GSM coverage is strong enough to establish a GPRS Internet connection.

To ensure a proper functioning of this system some peripherals are needed. As this system is to be used as a data logger, a Real Time Clock autonomous module ensures the correct time is never lost, and an SD card is used as an external data storage for back up purposes.

The power module present in this main sensor node is constituted by a battery pack and a DC/DC converter, and it is responsible for powering not only this sensor node but the whole network. Another proposed feature is the inclusion of a back-up battery for the computational system, which will allow the system to finish its operation when the main module runs out of electrical power.

As this main sensor node is to be deployed in an aquatic environment, precautions should be taken, i.e. it can be stored in an IP67 enclosure.

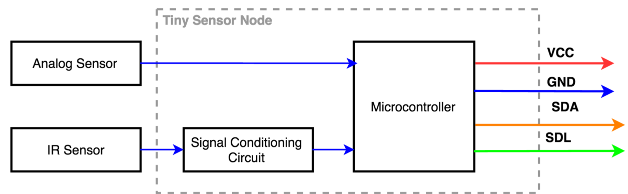

Figure 4. Tiny Sensor Node Architecture

In figure 4 it is possible to analyze the tiny sensor nodes functional architecture, where a microcontroller unit poses as the central point, as it ensures the execution of this device main tasks such as: reading the sensors, making the ADC conversion, and be part of an I2C based network.

One of the most important requirements for this device is its compatibility with a sensing method that allows the measurement of a mussel heartbeat signal, as in [8], therefore the tiny sensor nodes also includes the signal conditioning circuitry necessary for obtaining that results using an IR sensor.

These tiny sensor nodes work as I2C slaves that must be connected to a bus coordinated by the main sensor node, as it is possible to see in figure 2. As this devices must be able to operate in underwater conditions they are embedded in epoxy.

V. Prototype Implementation

After designing a solution, it was vital to implement a prototype that allows a practical validation of the proposed solution. This section focuses in the implementation of the two main functional blocks present in the proposed architecture.

A. *Main Sensor Node*

In the proposed architecture, the main sensor node is the key part of the whole system, as it is responsible for a considerable numbers of tasks, and for that a computational system is required to coordinate the sensor network, the collected data, and the communications.

Scientific literature on embedded systems describes several works based on embedded Linux boards, as Linux is a well suited operating system for the development of this kind of system, since it is possible to change to the software and adapt it to fulfil the specific needs of each application.

Therefore, a Beaglebone Black (BBB) single board computer running a Linux OS was chosen to be the main sensor node computational system, as it fulfils all the computational requirements for this specific application and it has great hardware interfacing capabilities, including two separate I2C buses.

The board used in this development has the Debian 7.9 distribution, a specifically customized version of Linux for the BBB, installed as its operating system. This Linux distribution has a built-in IDE, Cloud9, that can easily be accessed using Secure Shell (SSH) protocol, which was used to develop the applications responsible for ensuring the main sensor node operation.

C++ programming language was chosen for the development of the applications in charge of performing the main sensor node tasks, as this language allows object oriented programming, which is a suitable framework when programming this kind of devices.

In figure 5, it is possible to see a simplified architecture of the software implemented in the Beaglebone Black.

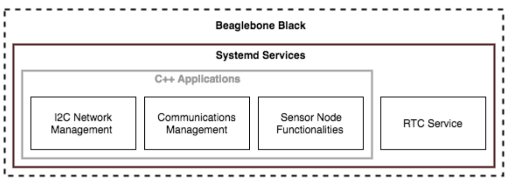

Figure 5. Main Sensor Node software architecture.

The developed applications were designed to be setup-orientated, operating with little intervention from the system user. This software architecture contemplates three different applications:

- I2C Network Management – Manages the whole I2C sensor network, sending sampling orders to the tiny sensor nodes present in either one of the two I2C buses available and retrieving the collected data.
- Communications Management – a simple program has been implemented to send the log files via email, using a secure simple mail transfer protocol (SMTP) server, to transmit the information to a previously specified address.
- Sensor Node Functionalities - Simple application to enable the utilization of the BBB as a sensor node itself, with analog sensors directly connected to the board.

In the Debian distribution running on the board, *Systemd* is a background process, a daemon that manages other daemons. It is the first daemon process to be executed on boot, and it is the last process to finish, before the board shutdown. *Systemd* is responsible for managing several services that control numerous processes in the BBB, as it allows parallel execution of services. Since this system is expected to have an autonomous and automated operation, several services were defined in *Systemd* to start on the Beaglebone start-up boot. This services ensured the periodic and coordinated execution of the developed applications that control the main sensor node tasks.

B. Tiny Sensor Node

Each tiny sensor node has to be able to process sampling orders sent by the main sensor node via I2C and to transmit the sampled data when the I2C master requests it. For this, it has to be able to read the sensors, do some signal conditioning if needed, make the ADC conversion, store the collected data, and be part of an I2C based network.

To ensure the execution of these tasks, it was vital to choose a microcontroller that fitted the requirements. Being so, a low cost and low power 8-bit microcontroller has been chosen, the ATtiny 85, produced by Atmel. Its manufacturer describes this device as a *"High Performance, Low Power AVR 8-bit Microcontroller"*. This IC has embedded ADC (10-bit resolution), which is crucial for data acquisition. Furthermore it allows the definitions of timers and Interrupt Service Routines (ISR), that are fundamental to control the sampling frequency, and it is also compatible with I2C protocol.

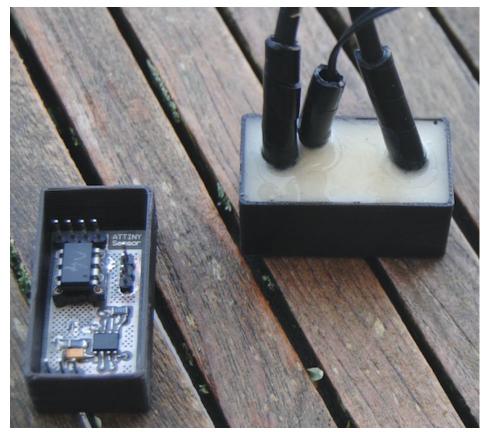

Figure 6. Tiny Sensor Nodes. (left: Tiny Sensor Node PCB. right: Waterproofed Tiny Sensor Node.)

The developed tiny sensor nodes were customized for a specific sensor and its conditioning circuit, as well as another arbitrary analog sensor, since it was one of the requirements of the project this system was being developed. However, this microcontroller firmware has been designed to handle two analog sensors, whatsoever is the sensor connected to it.

The firmware has been designed so that when an I2C master transmits the specified string to a specific slave, it executes the sampling order considering the designated pin, and stores the collected data. The program is also prepared to transmit the last sampling process information, whenever the main sensor node requests it.

Finally, after developing the firmware for the microcontroller and validating the conditioning signal circuit for the IR sensor, there was a need to develop a small sized print circuit board (PCB) to serve as a tiny sensor node that can be waterproofed using an epoxy resin in order to be used to collect underwater data, like it is possible to see in figure 6.

VI. System Operation

This section concentrates in how the implemented prototype must be used. This embedded sensor network is expected to have a straightforward operation that requires little maintenance and input from the user.

For putting this system up and running, it is just needed to insert an SD card in the Main Sensor Node containing a *'SETUP.txt'* file, where the sampling settings are defined for all the sensor nodes present in the sensor network. For each tiny sensor node connected to the I2C bus, that has a previously defined I2C address, there are two parameters that have to be defined: number of samples, sampling frequency at which the sample must be collected. It is also fundamental to specify a third parameter, the sampling process periodicity .

The main sensor node uses the information present in the setup file to create the sampling orders, to send each order to a specific tiny sensor node. Then, when a tiny sensor node receives a valid order, it executes the desired sampling. After the sampling process in the tiny sensor nodes is completed, and due to technical restrictions, the bus being shared by all the devices, the main sensor node will sequentially request each I2C slave the collected data. This process happens with the specified sampling process periodicity.

After each complete sampling process, the system tries to upload the log files containing the collected data to the predefined web mail account. If a valid Internet connection is available, the files are uploaded, if not, the system will try to upload all the non transmitted files after the next sampling process.

VII. Validation and Evaluation

The test and validation of the system is essential in order to evaluate if the features and architecture proposed are an interesting solution to solve the problem at hand. It was important to confirm that the system is able to operate autonomously for a considerable period of time, execute the defined sampling orders and collect underwater data.

A test was set up to validate the tiny sensor node board and its signal conditioning circuit, as well as its sampling capabilities. The diagram presented in figure 7 shows the setup used to realize this test.

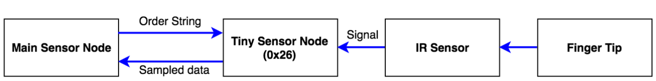

Figure 7. Heartbeat testbed.

In figure 8 it is possible to see the results obtained when using the board conditioning circuit to sample the heartbeat of an adult male. It was used with the CNY70 IR sensor in direct contact with a fingertip. The presented test is one of the tests, that allowed the validation of the signal conditioning circuit, the heartbeat signal has been sampled with a sampling frequency of 19.93 Hz.

The results obtained are coherent, as the circuit clearly detects the heartbeat peaks and the collected value can be processed in order to calculate the heartbeat frequency. This validates the tiny sensor node, as it executed the sampling

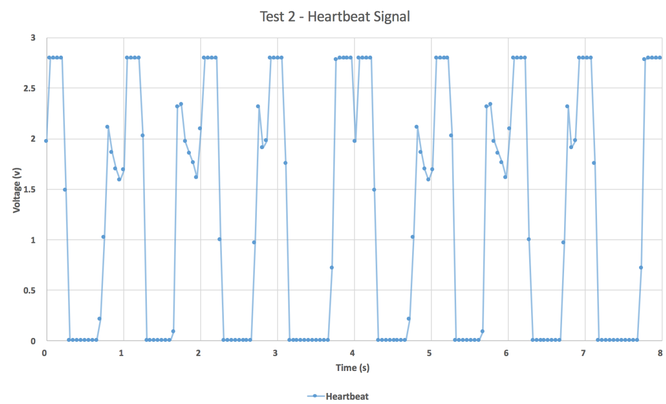

Figure 8. Sampled heartbeat signal of human adult male.

process as expected and the signal was properly filtered before the ADC conversion.

Another test was set up to have the system operating autonomously for a considerable period of time.

The testbed represented in figure 9, used two temperature sensors. One was connected directly to the BBB board, and the other was connected to an underwater tiny sensor node. That way it was possible measure the water temperature and the air temperature in the local the system is operating.

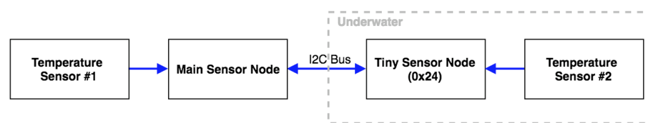

Figure 9. Testbed for temperature measurement test.

The sensor responsible for measuring the air temperature, had a sampling order defined in the main sensor node, to be executed by the Sensor Node Functionalities C++ application, that defined that the system was to collect 10 samples in 5 seconds (2Hz). With this data the main sensor node calculated the temperature average value at the time, and stored it a log file. Regarding the measurement of the water temperature, a similar sampling order was sent to the desired tiny sensor node, that executed the data collection, and retrieved the data when requested. That information was stored in a specific log file.

Then, the periodicity of both services that assures the cyclic execution of the Sensor Node Functionalities and I2C Network Management C++ applications are set to start every 10 minutes.

As it is possible to see in figure 10, the tiny sensor node used in this test was waterproof and was placed inside a container filled with water. In figure 11 it possible to see the results of this test. The main sensor node has been operating and collecting samples regarding the outdoors water and air temperature during a period of 55 hours. The obtained results are perfectly plausible, taking in consideration the local environment constraints.

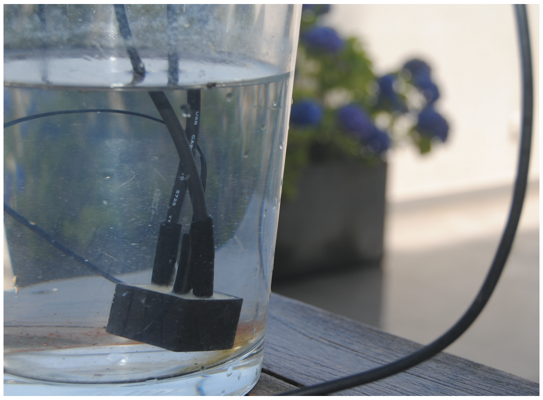

Figure 10. Underwater Tiny Sensor Node.

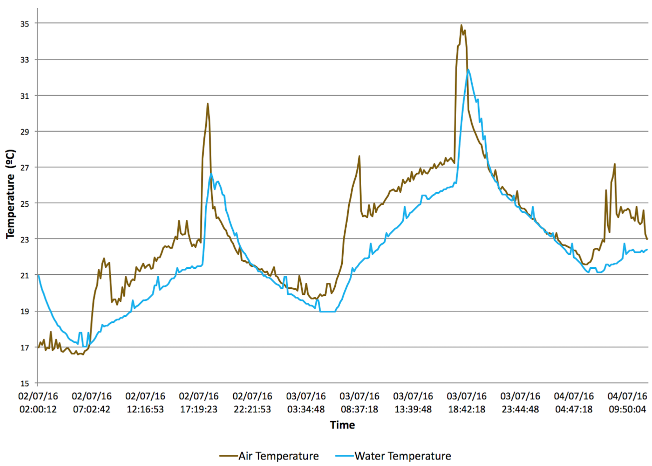

Figure 11. Underwater Tiny Sensor Nodes Test Results.

This test validates several features such as: the sensor node functionalities of the main sensor node, the underwater operation of the tiny sensor nodes, and the BBB autonomous operation for a long period of time, which is configured with the services defined in *Systemd*. This means the C++ applications responsible for the sampling processes defined in this test were running less than 30s in each 10 minutes.

VIII. Conclusions

A consistent architecture was proposed for a modular, extensible and reconfigurable system.

Currently, the two main functional blocks (Main Sensor Node and Underwater Tiny Sensor Nodes) and the communications between them (I2C Sensor Network) have been implemented and validated. The implemented prototype is able to operate autonomously and requires little input from its user.

The developed tiny sensor nodes were customized for a specific application that required a conditioning circuit. However, this microcontroller can be used with the developed firmware, in a smaller board if a different packaging of the ATtiny85 is used, interfacing with two arbitrary analog sensors. As it firmware has been designed so that when an I2C master transmits the specified string to a specific slave, it executes the sampling order considering the designated pin.

As expected, the I2C protocol proved to be a great choice as it removes any constraints in the sensor network topology design, and the possibility of connecting all the sensor nodes to a bus reduces drastically the cable cost in environments where several sensors are needed.

The developed system prototype has demonstrated this solution is of high interest, but there are still several features to be improved, however this system modularity eases further developments.

IX. Future Work

The following step for this system concerns its deployment and operation in the environment it was designed for. Being so, it is necessary to integrate the power module in the main sensor node and to prepare the I2C sensor network cables.

Another future development of great interest is this system integration with another parallel CIBIO/InBIO project, that contemplates the upload of sensor data collected by a Beaglebone Black to a relational database management system (RDMS), and its visualization in a website.

Acknowledgment

This work is financed by the ERDF – European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project ≪FCOMP-01-0124-FEDER-037281≫

References

[1] C. D. Whiteman, J. M. Hubbe, and W. J. Shaw, "Evaluation of an inexpensive temperature datalogger for meteorological applications," *Journal of Atmospheric and Oceanic Technology*, vol. 17, no. 1, pp. 77–81, 2000.

[2] J. Heidemann and R. Govidan, "An Overview of Embedded Sensor Networks," *Handbook of Networked and Embedded Control Systems*, pp. 1–20, 2004.

[3] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128601003024

[4] P. Blank, P. Kugler, D. Schuldhaus, and B. M. Eskofier, "A Data Collection and Communication Module for Telemedicine and mHealth Systems," *Proceedings of the 9th International Conference on Body Area Networks*, pp. 154–158, 2014. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-84936014076{\&}partnerID=tZOtx3y1

[5] A. Chianese, F. Piccialli, and G. Riccio, *Designing a Smart Multisensor Framework Based on Beaglebone Black Board*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 391–397. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-45402-2_60

[6] C. J.Manikandan and V.K.Agrawal, "Sensor monitoring using i2c protocol and labview," in *Proceedings of International Conference on "Emerging Research in Computing, Information, Communication and Applications*, 2013.

[7] M. Gandra, R. Seabra, and F. P. Lima, "A low-cost, versatile data logging system for ecological applications," *Limnology and Oceanography: Methods*, vol. 13, no. 3, pp. 115–126, 2015. [Online]. Available: http://dx.doi.org/10.1002/lom3.10012

[8] N. Burnett, R. Seabra, M. De Pirro, D. Wethey, S. Woodin, B. Helmuth, M. Zippay, G. Sarà, C. Monaco, and F. Lima, "An improved noninvasive method for measuring heartbeat of intertidal animals," *Limnology and Oceanography: Methods*, vol. 11, no. FEB, pp. 91–100, 2013.

[9] *I2C manual*, Philips Semiconductors, 2003.