Hindawi Publishing Corporation
Journal of Sensors
Volume 2014, Article ID 125874, 11 pages
http://dx.doi.org/10.1155/2014/125874

Research Article

FPGA Based Single Chip Solution with 1-Wire Protocol for the Design of Smart Sensor Nodes

M. D. R. Perera,[1] R. G. N. Meegama,[1] and M. K. Jayananda[2]

[1] *Department of Computer Science, University of Sri Jayewardenepura, 10250 Nugegoda, Sri Lanka*
[2] *Department of Physics, University of Colombo, 00300 Colombo 03, Sri Lanka*

Correspondence should be addressed to R. G. N. Meegama; rgn@sci.sjp.ac.lk

Received 27 May 2014; Revised 26 October 2014; Accepted 30 October 2014; Published 23 November 2014

Academic Editor: Stefania Campopiano

Applications that involve monitoring of water quality parameters require measuring devices to be placed at different geographical locations but are controlled centrally at a remote site. The measuring devices in such applications need to be small, consume low power, and must be capable of local processing tasks facilitating the mobility to span the measuring area in a vast geographic area. This paper presents the design of a generalized, low-cost, reconfigurable, reprogrammable smart sensor node using a ZigBee with a Field-Programmable Gate Array (FPGA) that embeds all processing and communication functionalities based on the IEEE 1451 family of standards. Design of the sensor nodes includes processing and transducer control functionalities in a single core increasing the speedup of processing power due to interprocess communication taking place within the chip itself. Results obtained by measuring the pH value and temperature of water samples verify the performance of the proposed sensor node.

1. Introduction

With rapid advances in modem science and technology, the concept of sensor networks is gaining ground in a variety of fields, especially in control and systems science. A sensor node in a sensor network is an instance of a sensor network application that comprises one or more sensors and actuators to achieve a specific monitoring demand. Such a sensor node consists of a processing unit to control and process raw data collected from sensors, a transceiver to transfer data and commands with other sensing nodes, a memory unit to keep the transducer electronics data sheets, sensor data, and a power source to provide energy.

When designing a sensor node for monitoring applications, the primary requirements are low equipment cost, low power consumption, small size, and wireless communication. When a node is to be set up in a remote location, providing power is a challenging task in wireless sensor networks. As such, chips and sensors that conserve power during processing functions in the node use wireless communication protocols and equipment to ensure reliable communication, save power by triggering sleeping/wake-up modes, and use rechargeable batteries with the energy harvesting method (solar or wind) to improve the lifetime of a sensor node [1–3].

As most applications involving sensor networks operate in natural environments that exhibit dynamic behavior, it is often necessary to reconfigure the nodes to comply with the environment. Although network and sensor node failures may occur under a natural environment, a single node failure should not affect the entire network. It should be able to readjust communication within the network with remaining nodes and should have the ability to add a new node to the network or remove an existing node from the network without affecting the whole system. Hence, the processing chip in the sensing modules must provide enhanced flexibility for the designer to implement a particular algorithm to accommodate this dynamic behavior.

Ultralow power microcontrollers, having limited computing power, are used to design wireless sensor nodes whereas FPGAs are used to achieve high performance while consuming low power [4, 5]. A combination of an FPGA and a microcontroller can be used as a programmable chip to develop a sensor node in which the FPGA is used to process digital signals while the microcontroller handles

communication and control [5–7]. In contrast, the technique presented in this paper implements a sensor node that includes communication modules handling, data processing, and transducer control functionalities in a single FPGA chip.

2. Theoretical Background

In sensor network applications, a sensor node is referred to as a Transducer Interface Module (TIM). Transducers are interfaced with a TIM to provide functions such as self-identification, self-diagnostics, self-description, location-awareness, time-awareness, data processing, reasoning, data fusion, and alert notification, which results in a "smart" device [8, 9]. As the proposed sensor node has these capabilities, it is called a smart sensor node.

The IEEE 1451 family of standards defines a set of open, common, network-independent communication interfaces for connecting transducers (sensors or actuators) having different kinds of communication protocols. It provides implementation standards of Transducer Electronic Data Sheets (TEDS) that stores transducer identification, calibration, data correction, measurement range, manufacturer-related information, and so forth, to facilitate a common message and commands structure to connect transducers with systems or networks via a wired or wireless media [8].

The IEEE P1451.0 standard defines a set of common commands, common operations, and TEDS' formats for the family of IEEE 1451 standards, allowing access to any transducer in the 1451-based wired and wireless networks. It provides a basic guideline for other standards of the 1451 family and future members [8, 9].

In the design stage, FPGA chips are used to implement TIM because of their higher performance, flexibility, reprogrammability, and reconfigurable capabilities according to dynamic changes in the environment and, at the same time, they consume low power [4, 5, 10].

The IEEE 802.15.4 (ZigBee) standard has a 250 kbps transmission rate with three different frequency bands and an average range of 100 meters. It supports mesh topology to connect nodes and can contain up to 65,536 nodes on a single network with low energy consumption. As such, it has become a popular wireless communication method in sensor network applications as a low cost and a reliable method with advanced network capabilities [1, 11–14].

3. Related Works

Zennaro et al. [1] have presented the design of a wireless sensor network to measure and monitor water quality. As a manual sampling method often causes considerable delays that hamper early warning and appropriate treatments, designers had to pay more attention to local conditions before implementing the wireless sensor network in a challenging environment. They have contributed two main challenge areas: energy consumption and internetworking. The wake-up mechanism, which triggers sleeping/wake-up modes, and ZigBee protocols were used to reduce energy consumption.

Zhiyong et al. [6] have presented a novel architecture for a wireless vision sensor node based on a low-cost, low-power FPGA and microcontroller system on a programmable chip. They have managed to lower the power consumption by using a sensor node having high computational capability and high flexibility to support various applications in different environments where the FPGA is utilized to implement complex and parallel JPEG image compression algorithms.

A framework based on a modular architecture for situation-based reconfiguration in a wireless sensor network (WSN) is proposed in [15]. These WSNs, when used in complex operating environments, need powerful microprocessors to perform processing tasks. As such, FPGAs have been used for on-board processing in order to overcome this barrier. In this context, the parallel modular FPGA-based design seems to have gained more performance, flexibility, and lower power consumption than microprocessor-based systems.

O'Flynn et al. [16] have presented the development of a modular and miniaturized wireless platform for sensor networks where the sensor node is designed in a modular nature providing enhanced flexibility and power-efficiency by reducing the size of a module to a a 25 mm cube. The communication module mainly consists of a microcontroller and a radio frequency (RF) transceiver to establish communication between the sensor nodes. The processing module, designed using an FPGA chip, carries out the digital signal processing tasks while the sensing module acquires readings from different sensors. The FPGA and sensor module can be customized according to end-user requirements.

Postolache et al. [17] have presented the design and implementation of an *in situ* water quality calibration system using a reconfigurable FPGA that controls different actuators (pumps and electrovalves) and data acquisition tasks from different water quality sensors. A real-time controller is used to process real-time water quality data whereas a wireless data communication media transmits such collected data.

Rasin and Abdullah [12] have presented a ZigBee based wireless sensor network to monitor the quality of water. A low cost, flexible, fast, and reconfigurable sensor node for real-time monitoring and controlling of multisensors in an application to secure food is also revealed in [18].

4. Methodology

This paper presents the implementation of a single chip solution embedding communication, processing, and transducer control functionalities in the FPGA in accordance with the IEEE 1451 standard of interfacing transducers to a network.

Figure 1 shows the basic block diagram of the transducer interface module (TIM) designed using an FPGA and having a number of functional blocks such as the Universal Asynchronous Receiver and Transmitter (UART) interface block to handle all transmissions and receptions, a memory block to maintain the required TEDS and sensor readings, a main controller block to handle the IEEE 1451.0 service, a transducer controller block, and a transducer interface block to take care of all functionalities of the transducers. A transducer specific

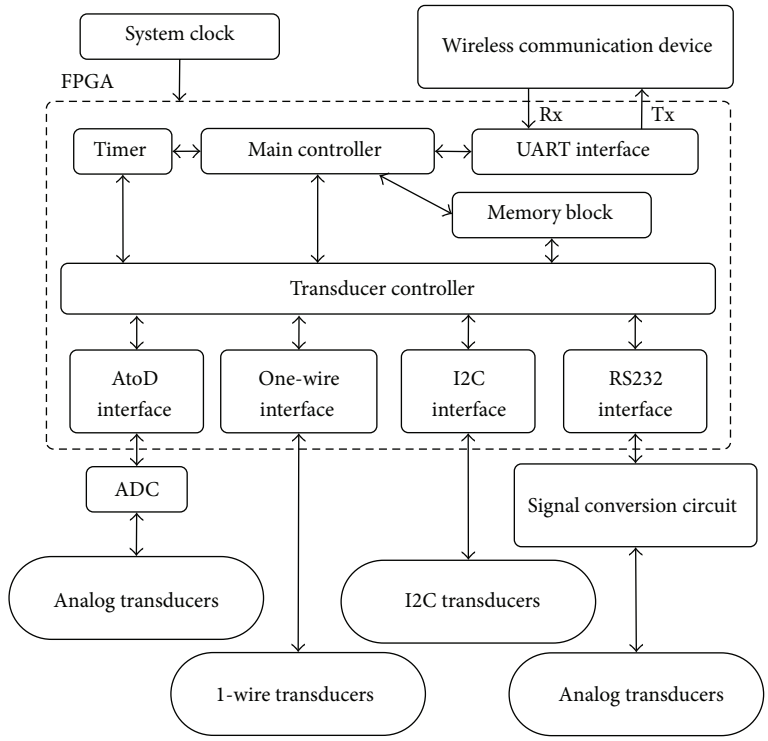


Figure 1: Internal architecture of the TIM showing the functional blocks with interconnectivity and transducer interfaces.

communication protocol is implemented in the transducer interface blocks, such as analog to digital, RS232, and 1-wire digital sensor interface, to connect transducers [19, 20].

4.1. UART. A wireless communication module is connected to the TIM through a UART interface. During wireless communication, malicious bytes may be generated due to inherent noise present in the operating environment. To mitigate such malicious error "start" and "terminate" characters are embedded into each message and an additional functional block, called a command filter, is introduced to the UART. The command filter identifies the received byte and if it is a starting character, it allows the receiving data buffer to store the next reserving bytes of the message until it receives the terminating character of the message. The filter unit does not release the message if the starting and terminating characters are not properly received. Therefore, the proposed methodology minimizes errors that might occur due to such invalid commands. At every new command, the command filter resets the Rx buffer to cater to the new message. In the design presented in this paper, a single starting bit, eight data bits, a no parity bit, and a single stop bit communication protocol are used [21].

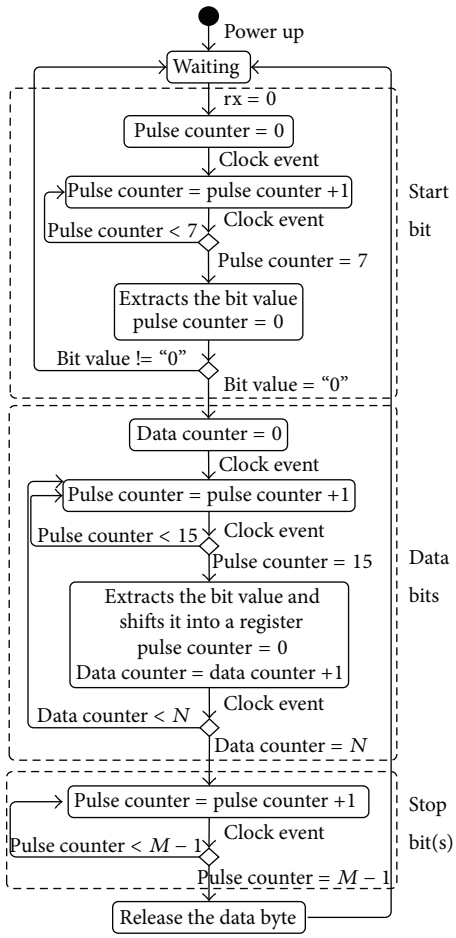
The transmitter circuit simply sends out data at a predefined baud rate while the receiver synchronizes the data stream using predetermined parameters. To avoid aliasing, improve resolution, and reduce noise, an oversampling scheme is used to extracts the bit value of the signal received. A rate of 16 times the baud rate is the most widely used for oversampling frequency in the design of the UARTs [21–23]. Figure 2 illustrates the steps required to extract the data byte from the receiving signal. The oversampling frequency required to sample the input signal is generated by dividing the system clock by the frequency divider. The number, by which the system clock must be divided, is determined by taking the ratio of the system clock speed to 16 times the baud rate; that is, divider = frequency of the system clock in Hz/(baud rate × 16).

The time gap between two request messages is required to avoid signal interference. The minimum time required to perform a single data cycle, T_M, is defined as

$$T_M = \left[\frac{\text{LDF}}{(\text{baud rate})} \times \left(N_{\text{REQ}} \times N_{\text{RES}} \right) \right] + T_P, \quad (1)$$

where T_P is data processing time, N_{REQ} is number of data frames in the request message, N_{RES} is number of data frames in the response message, and LDF is length of the data frame.

4.2. Memory Block. The design of the memory block, as depicted in Figure 3, includes a ROM that keeps the necessary TEDS of the TIM and a RAM for temporarily holding the current readings of the sensors. The "block" and "interleave" methods are the two techniques that are used to combine the data sets. The interleave method requires all data blocks to be of the same size. However, the length of the data sets in every transducer is not the same [8]. Two separate memory tables are maintained to identify the starting position of the stored record in both the RAM and the ROM. The cells having an 8-bit data width are used to design the memory blocks in the FPGA.



M = 16, 24, and 32 pulses for 1, 1.5, and 2 stop bits, respectively
N = number of data bites

Figure 2: Internal logic of the receive block in the UART.

As defined in the IEEE 1451.0 standard, four TEDS are required for all the TIMs, such as Meta TEDS, Physical TEDS, and User Transducer Name TEDS, and Transducer Channel TEDS. These are mandatory TEDS while others, such as Calibration TEDS, Frequency Response TEDS, Transfer Function TEDS, Text TEDS, End User Application Specific TEDS, and Manufacturer Defined TEDS, are optional.

The TEDS ROM is designed in order to provide separate memory blocks for each transducer to keep their respective TEDS where all TEDSs relevant to a particular transducer are defined within the allocated memory block [8, 9, 24]. The size of the ROM block depends on the number of TEDS and the size of each TEDS. S_{ROM} is defined as the minimum length of the ROM to implement necessary TEDS in the TIM and is given as

$$S_{\text{ROM}} = \sum_{j=1}^{m}\sum_{i=1}^{n}\left[\text{LF}_{ij} + 6\right], \tag{2}$$

where LF_{ij} is number of octets in the ith field of the jth TEDS.

As mentioned above, the RAM receives separate memory blocks for each transducer to keep its sensor readings. The minimum size of the RAM is defined as

$$S_{\text{RAM}} = \sum_{i=1}^{n}\text{LSR}_i, \tag{3}$$

where LSR_i is the number of octets in the ith sensor.

4.3. Sensor Interface Blocks. The sensor interface blocks, such as analog-to-digital (A to D), RS232, and 1-wire interfaces, are implemented to interfaces the sensors that have different transducer specific communication protocols. The implementation of these sensor interfaces is shown in Figure 4.

The A to D interface is used to interface any kind of analog sensors to the FPGA through an A to D conversion circuit. The sensor reading time, T_{ADC}, depends on the conversion delay (T_{CNV}) as

$$T_{\text{ADC}} = T_{\text{CNV}}. \tag{4}$$

The RS232 sensor interface block is derived using the UART interface where the necessary sensor specific commands are included in the ROM. Based on these requirements, the transducer controller executes the commands to perform the functionalities of the sensor.

In digital sensors having a 1-wire protocol, data are transferred serially using a single data line with a ground reference that can be used to connect one or more slave devices to the 1-wire bus. The 1-wire master device communicates with a slave device using a hard-coded 64-bit identification number and utilizes a self-timer driven by an internal oscillator that is synchronized with the falling edge of the master [25]. All 1-wire operations are performed using four signaling modes as follows:

(1) reset sequence with reset pulse and presence pulse;

(2) write "1": send a "1" bit value to 1-wire slaves;

(3) write "0": send a "0" bit value to 1-wire slaves;

(4) read bit: read a bit from 1-wire slaves.

Figure 5 gives the implementation of the above four signaling modes by the master device. The 1-wire protocol is a 3-phase transaction that begins with a master reset followed by device selection commands (ROM level commands) and a device command (device level command) that are executed to access internal registry values of the sensor. Figure 6 illustrates the execution sequence of the command to read a temperature value from the DS18B20 1-wire digital thermometer.

The time interval required to take a reading from the sensor is given by

$$\begin{aligned} T_{\text{1-wire}} = {} & \sum_{i=1}^{n}\left[\text{LC}_i \times \left(T_{\text{RW}} + T_{\text{RCV}}\right)\right] + m \times T_{\text{RST}} + T_{\text{CNV}} \\ & + \left[\text{BL} \times \left(T_{\text{RW}} + T_{\text{RCV}}\right)\right], \end{aligned} \tag{5}$$

where LC_i is the number of bits in the ith command, T_{RW} is time slot which define for read or write bit operations, T_{RCV}

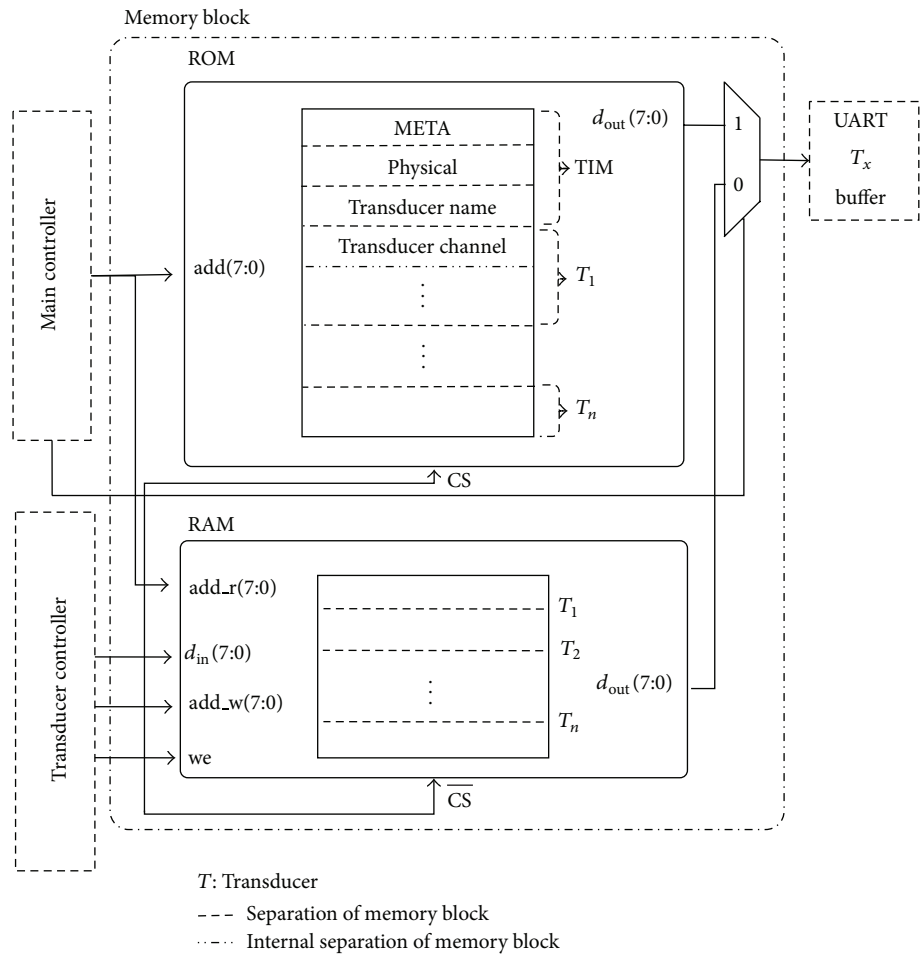


Figure 3: The internal memory organization of ROM that keeps TEDS and RAM holding sensor readings.

is recovery time between the two time slots, T_{CNV} is data conversion delay, m is number of reset pulse in the command sequence, and BL is number of bits in the sensor reading.

4.4. Transducer Controller. In the proposed architecture, a transducer controller is used to control the sensor interface blocks and perform signal conditioning and data conversion of the transducers. Periodically, the controller instructs the sensor interface blocks to collect the most recent readings from the sensors and updates the allocated place in the RAM in the memory block.

4.5. Main Controller. The main controller block is responsible for handling the IEEE 1451.0 service and takes care of all functionalities of the TIM. After reception of the entire request message, the UART notifies the main controller to proceed. The main controller then decodes the message and identifies the received command using the IEEE 1451 command standard. If it is a valid command, it executes the process and constructs the reply message; otherwise, an error message is generated. Finally, the reply message is loaded into the transmission buffer for subsequent transmission. The main controller accesses the RAM to obtain the most recent sensor values and accesses the ROM to obtain transducer information stored as a TEDS. The state diagram of the TIM is illustrated in Figure 7.

5. Experimental Setup

The application of wireless sensor networks for water quality monitoring under the domain of monitoring environmental parameters is used to evaluate the proposed sensor nodes under laboratory conditions. Two water quality parameters, namely, the temperature and the pH value, are selected for this purpose.

The TIM is implemented in a Digilent Basys2 development board having a XC3S250E FPGA. The Xilinx ISE Design Suite 14.1 was utilized as the developing tool with VHDL as the programming language.

In order to read the sensor values from the TIM, a "read transducer channel" command is sent to the TIM and the sensor values are extracted from the reply message. Figure 8 illustrates how external devices such as a ZigBee communication module, 1-wire digital thermometer (DS18B20), and a pH sensor kit are connected to the BASYS2 FPGA board via the UART, one-wire, and RS232 interfaces, respectively, within a

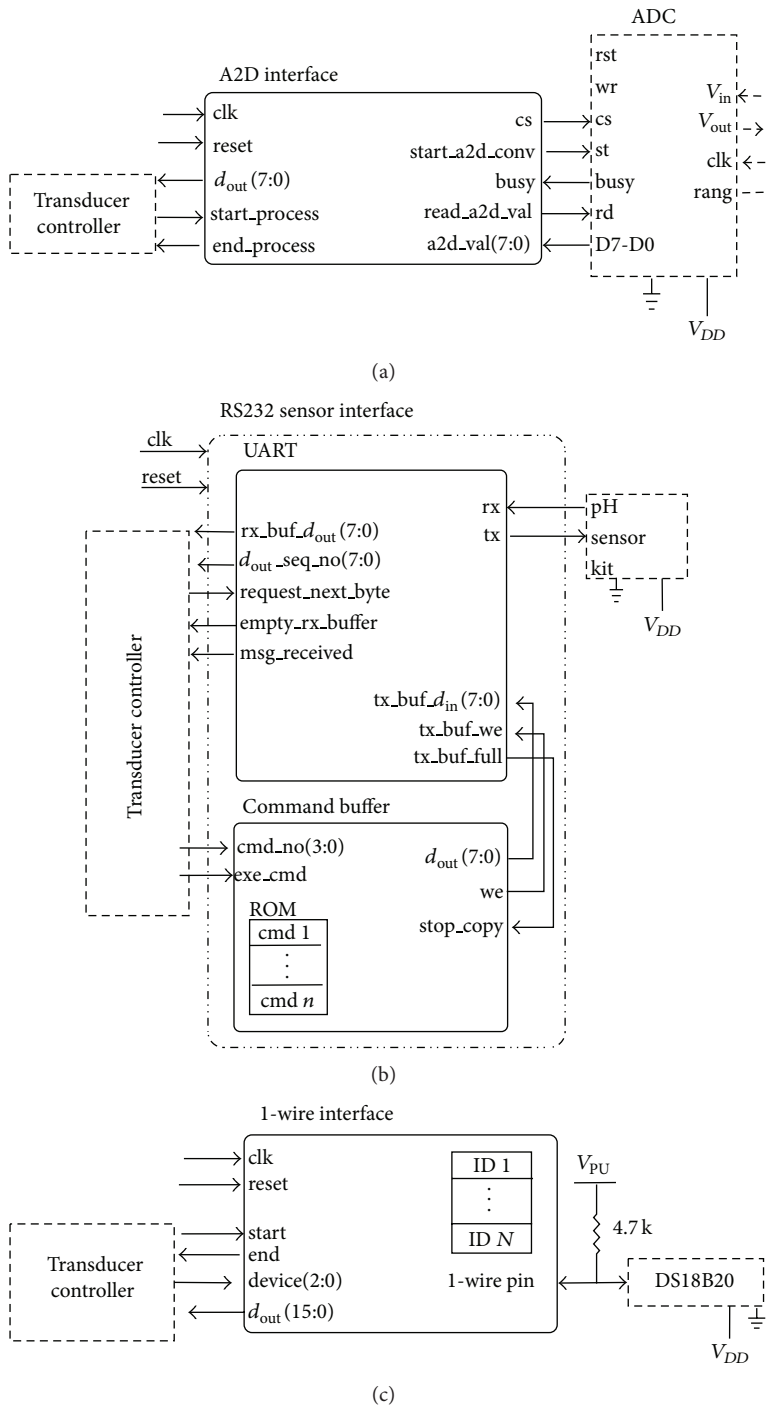


Figure 4: Sensor interface blocks used to connect transducers with different types of protocol: (a) analog to digital interface, (b) RS232 interface, and (c) 1-wire interface. *Note*. The blocks shown in the dashed lines are not part of the sensor interfaces.

compact space. As seen, three AA batteries are used to power up the devices.

The DS18B20 waterproof digital thermometer has an accuracy of ±0.5°C with a 0.0625°C resolution connected to the TIM through the 1-wire interface [25]. Although it manages to obtain the readings up to 4 decimal points, the final result is rounded up to one decimal point which is the standard acceptable accuracy for monitoring environmental water quality parameters [26]. Hach's sensION + EC5 portable meter (http://www.hach.com/) having an accuracy of ±0.2°C with one decimal point resolution (0.1°C) is used as a standard reference.

An Atlas pH meter, having hardware/firmware v4.0 (https://www.atlas-scientific.com/), was connected to the

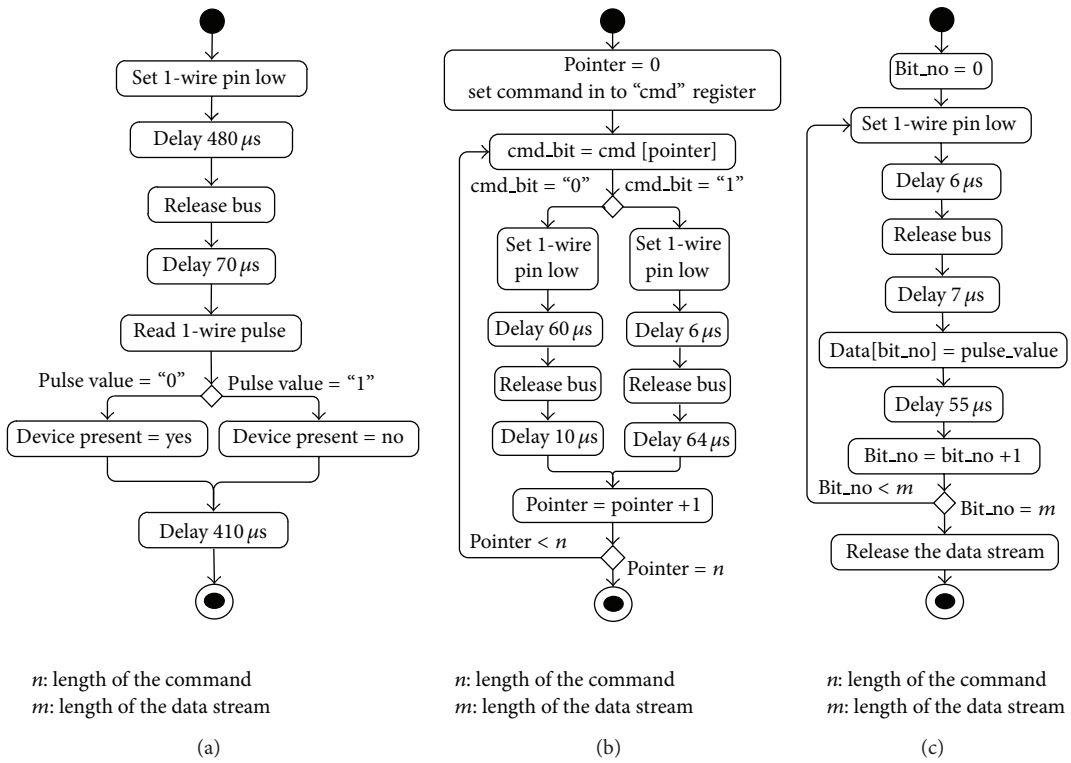


Figure 5: The basic functional steps of the 1-wire communication handled by the master device: (a) reset slaves, (b) write a data stream to slave device, and (c) read data stream from slave device.

TIM through the RS232 interface. The Hach's HQ40d portable meter with software version 2.2.0.721, 2.0.0.699 was used as a standard reference. Prior to interfacing the Atlas pH meter to the TIM, both pH meters were calibrated using the same pH solutions.

By varying the acidity levels, 21 different samples of water are collected as in Table 1. Subsequently, the pH value of each sample is taken using both the sensor and the standard probe.

6. Results and Discussion

6.1. UART. In order to test the stability of the UART, a command is used to query the TEDS that are built into the FPGA memory block. A "command message" is transmitted to the FPGA embedding the "Read TEDS Segment Command" [8]. After receiving the request message, the controlling block executes the command and constructs a reply message by embedding a "TEDS' segment command reply" structure in the reply-dependent octets section [8].

The computer checks whether the command flows smoothly or not and calculates the message loss rate in different baud rates under practical environmental conditions. All request and response messages as well as "read TEDS segment" command and "reply TEDS segment" commands are constructed in accordance with the IEEE 1451.0 standard. The data to be tested is transmitted through the UART repeating 1000 times in a selected baud rate. At the next step, the baud rates are varied and the stability of the TIM in different baud rates is verified.

The minimum time required to perform a single data cycle, T_M, is rounded up to the nearest 10th value (T_T) and it is defined as the time interval between two request messages for a particular baud rate. Baud rates such as 2400, 4800, 9600, 14400, 19200, 38400, 56000, 57600, and 115200 are selected to test the stability of the UART. The experiment is carried out 1000 times for each baud rate and the percentage data loss in the experiment is measured. It is observed that the UART interface performs steadily in all tested baud rates except at 4800 and 115200.

In practice, proper implementation of several baud rates is a tedious task. For instance, it is difficult to find a suitable baud rate devisor integer for the 4,800 baud rate. Although, theoretically, the nearest integer values are 652 or 651, the loss rate is around 7%. When the devisor is changed to 650, the loss rate is reduced up to 1% whereas in a 115,200 baud rate, the data loss percentage is around 6.4%. Any deviation in the results is not observed when the test is carried out in different time intervals by increasing the time per one testing cycle for the baud rates as mentioned above. As such, this oversampling scheme may not be appropriate for high data rates because a slight change may cause a big difference, which results in higher error rates.

6.2. Sensor Interfaces. The temperature values are obtained from the reference meter and the DS18B20 sensor. The temperature measured using the sensor (T) may not represent the actual temperature at a particular place because every sensor has its own measurement error (δt) as defined by the

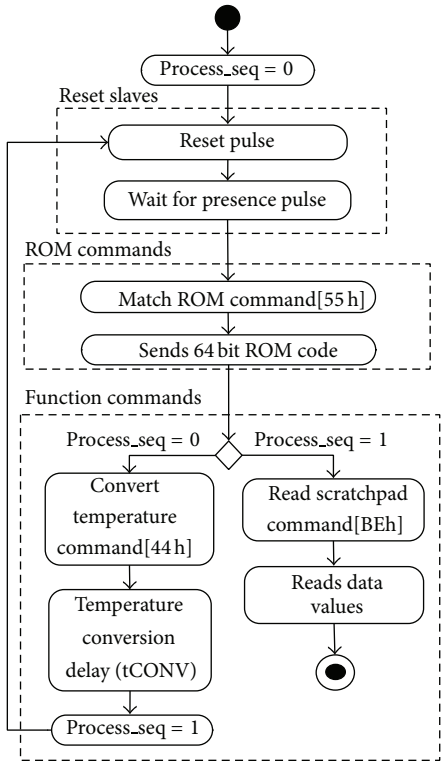


Figure 6: Functional steps required to read the temperature value from the DS18B20 sensor.

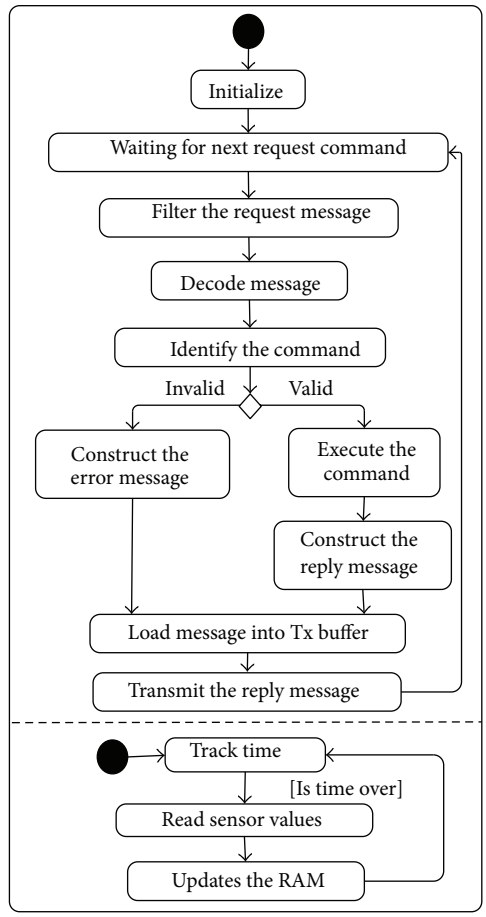


Figure 7: The operation of the TIM implemented in the FPGA.

manufacturer. Therefore, the actual temperature is between $(T + \delta t)$, the upper bound, and $(T - \delta t)$, the lower bound. The graph in Figure 9 displays possible temperature ranges in which the actual temperature may occur. According to this graph, the temperature ranges of both the DS18B20 and the sensION meter overlap in every time interval verifying the accuracy of the DS18B20 digital thermometer that is connected to the TIM.

Table 1 shows the pH values taken from 21 different samples. The pH value of a fresh water sample is 6.85 when measured using an HQ40d meter and 6.92 in an Atlas meter. A paired sample t-test confirms that there is no significant difference between the means of the HQ40d meter and the Atlas meter at 5% level of significance (P value = 0.464).

6.3. Evaluation of the TIM. A Xilinx Spartan-3E family FPGA is used to implement the TIM that requires a gate capacity of at least 250 K. Table 2 lists the synthesis report and presents the hardware utilization of the proposed architecture as generated from the Xilinx tool. The total power consumption of the TIM is dependent on several factors such as the type of the development board, the number of sensors, the communication device, and peripherals connected. A spreadsheet-based power estimation tool (XPE) provided by Xilinx is used to preestimate the power consumption of the design by

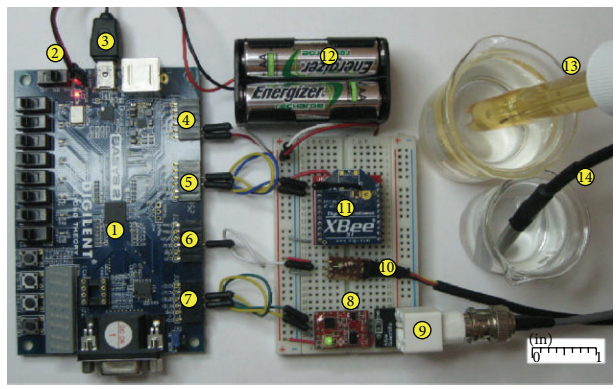


Figure 8: Experimental setup of the sensor node: (1) Xilinx XCS250E FPGA, (2) cables to power the FPGA board, (3) USB programming cable, (4) cables to power the FPGA board sensors, (5) UART interface to connect XBee device, (6) 1-wire interface, (7) RS232 sensor interface to connect the pH kit, (8) Atlas Scientific pH circuit, (9) BNC connector, (10) DS18B20 sensor connector, (11) XBee device, (12) battery pack, (13) Atlas Scientific pH probe, and (14) waterproof DS18B20 temperature sensor.

Table 1: pH readings taken from HQ40d meter and Atlas pH meter at different dilutions.

Sample number	Amount of water (mL)	Amount of acidity (H_2SO_4) (mL)	Amount of alkalosis (NaOH) (mL)	HQ40d meter	Atlas pH meter
1	100	—	5 of 0.1 M	11.80	11.89
2	100	—	4 of 0.1 M	11.69	11.72
3	100	—	3 of 0.1 M	11.60	11.63
4	100	—	2 of 0.1 M	11.40	11.43
5	100	—	1 of 0.1 M	11.00	11.04
6	100	—	5 of 0.01 M	10.54	10.58
7	100	—	4 of 0.01 M	10.38	10.43
8	100	—	3 of 0.01 M	10.28	10.35
9	100	—	2 of 0.01 M	10.06	9.98
10	100	—	1 of 0.01 M	9.37	9.32
11	100	—	—	6.85	6.92
12	100	1 of 0.01 M	—	5.74	5.79
13	100	2 of 0.01 M	—	4.02	3.93
14	100	3 of 0.01 M	—	3.65	3.56
15	100	4 of 0.01 M	—	3.48	3.41
16	100	5 of 0.01 M	—	3.34	3.28
17	100	1 of 0.1 M	—	2.85	2.80
18	100	2 of 0.1 M	—	2.53	2.47
19	100	3 of 0.1 M	—	2.41	2.37
20	100	4 of 0.1 M	—	2.29	2.23
21	100	5 of 0.1 M	—	2.20	2.14

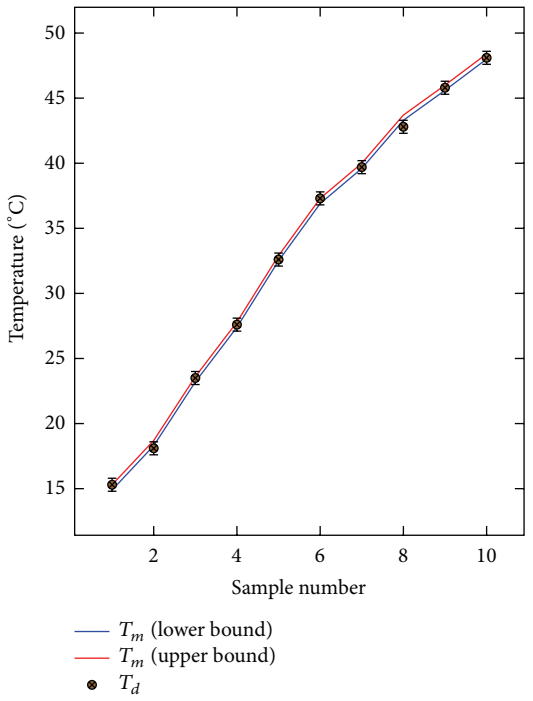


Figure 9: Actual temperature ranges measured using DS18B20 waterproof digital thermometer and sensION meter.

Table 2: Summary of device utilization as generated by the Xilinx tool.

Logic utilization	Used	Available	Utilization
Total number of slice registers	2,471	4,896	50%
Number of 4 input LUTs	2,481	4,896	50%
Number of occupied slices	2,389	2,448	97%
Total number of 4 input LUTs	2,574	4,896	52%
Number of bonded IOBs	8	92	8%
Number of BUFGMUXs	2	24	8%

importing data from the ISE. The XC3S250E FPGA consumes 51 mW of static power [5, 27] and it varies dynamically with the design and the frequency of the clock input. The graph in Figure 10 illustrates the estimated power consumption of the proposed design at different clock frequencies using different chips in the Spartan-3E family. During all calculations, the ambient temperature is maintained at 25°C.

A linear relationship is observed between the power consumption and the frequency for a selected set of FPGA chips as shown in Figure 10 where the rate of power consumption with respect to the frequency increases based on the capacity of the chip. These results, together with the estimated values of the intercept and the gradient for each chip that are listed in

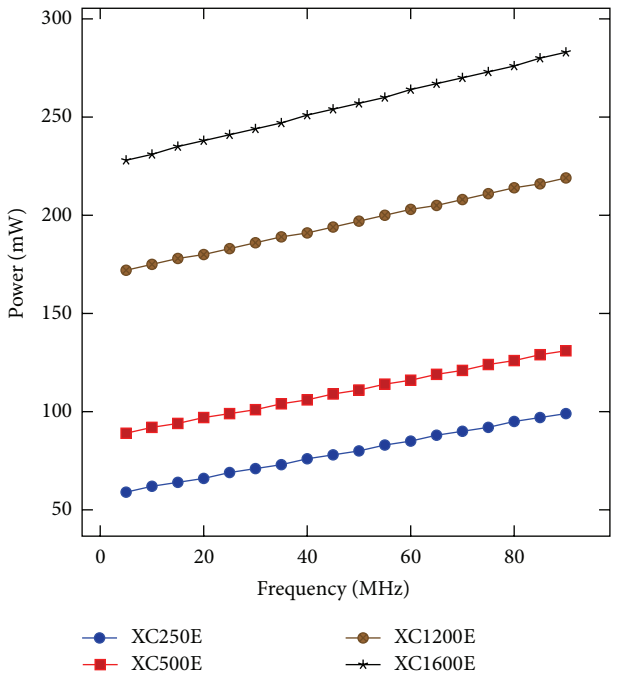


Figure 10: Estimated power consumption using several FPAG chips in the Sparten 3E family under different clock frequencies.

Table 3: The intercept and gradient of selected FPGA chips.

Device	Intercept	Gradient (10^{-3})
XC250E	56.9	472
XC500E	86.7	494
XC1200E	169.0	553
XC1600E	225.0	645

Table 4: Power consumption of the experimental setup.

Status of the node	Voltage (V)	Current (A)	Power (mW)
Static mode	4.19	0.11	460.9
Operation mode	3.78	0.13	491.4

Table 3, imply the ability to save power by selecting a suitable chip having a minimal capacity.

Under actual practical operating conditions, the total power consumption of the design, as illustrated in Figure 10, is calculated using the voltage supplied and the gain current. Measurements are taken at two different conditions, such as before downloading the design into the FPGA (static mode) and during the operation time of the TIM (operation mode). These readings are listed in Table 4 that indicates a 30.5 mW power consumption by the TIM. In this setup, 50 MHz is used as the input clock frequency of the FPGA. Notice that Actel has developed ultralow-power FPGAs particularly in the IGLOO family consuming power in μW level (between 2 μW and 5 μW) [5]. Such ultralow-power FPGAs are able to reduce the power consumption of the TIM significantly. Table 5 shows the list of equipment with associated costs used for the experimental setup. The cost can be reduced further by using basic FPGA boards (e.g., a Core3S250E board, which has in-built XC3S250E FPAG at 27.99 US$ [28]) or a custom-designed board. At this point, it is worth mentioning that O'Flynn et al. [16] have also presented a modularized platform with a 25 mm cube structure for the design of the sensor nodes.

Table 5: Equipment cost of the experimental setup.

Sample number	Price (US$)
Digilent development board	69.00
Atlas pH sensor kit	60.00
Atlas EC sensor kit	105.00
Atlas DO sensor kit	160.00
DS18B20 sensor	3.8
Digi XBee device	22.95
Battery pack	5.3
Consumables (resisters, wires, etc.)	1
Total	427.05

7. Conclusions

The proposed single chip solution to interface transducers to sensor networks using FPGAs can be easily replaced with a wired or wireless method, by using a wireless module having a UART interface. Furthermore, it is possible to design separate RS232 interfaces according to some specific required baud rates. However, before selecting a particular baud rate, the TIM must be tested under practical conditions without solely depending on mathematical calculations.

Any sensor having a 1-wire protocol can easily be connected through the 1-wire interface in the TIM. As such, it is possible to implement standard or customer specific digital protocols in the FPGA to interface transducers to sensor networks without changing the designed architecture.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This research work is funded by the University of Sri Jayewardenepura Research Fund under Grant no. "ASP/06/RE/SCI/2012/06."

References

[1] M. Zennaro, A. Floros, G. Dogan et al., "On the design of a Water Quality Wireless Sensor Network (WQWSN): an application to water quality monitoring in Malawi," in *Proceedings of the 38th International Conference Parallel Processing Workshops (ICPPW '09)*, pp. 330–336, Vienna, Austria, September 2009.

[2] F. Xia, "Wireless sensor technologies and applications," *Sensors*, vol. 9, no. 11, pp. 8824–8830, 2009.

[3] S. Sudevalayam and P. Kulkarni, "Energy harvesting sensor nodes: survey and implications," *IEEE Communications Surveys and Tutorials*, vol. 13, no. 3, pp. 443–461, 2011.

[4] J. Valverde, A. Otero, M. Lopez, J. Portilla, E. de la Torre, and T. Riesgo, "Using SRAM based FPGAs for power-aware high performance wireless sensor networks," *Sensors*, vol. 12, no. 3, pp. 2667–2692, 2012.

[5] A. de la Piedra, A. Braeken, and A. Touhafi, "Sensor systems based on FPGAs and their applications: a survey," *Sensors*, vol. 12, no. 9, pp. 12235–12264, 2012.

[6] C. H. Zhiyong, L. Y. Pan, Z. Zeng, and M. Q.-H. Meng, "A novel FPGA-based wireless vision sensor node," in *Proceedings of the IEEE International Conference on Automation and Logistics (ICAL '09)*, pp. 841–846, Shenyang, China, August 2009.

[7] K. Shahzad, P. Cheng, and B. Oelmann, "Sentiof: an FPGA based high-performance and low-power wireless embedded platform," in *Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS '13)*, pp. 901–906, September 2013.

[8] IEEE, "Standard for a smart transducer interface for sensors and actuators—common functions, communication protocols and transducer electronic data sheet (teds) formats," IEEE Standard 1451.0, 2007.

[9] E. Y. Song and K. Lee, "Understanding IEEE 1451—networked smart transducer interface standard—what is a smart transducer?" *IEEE Instrumentation and Measurement Magazine*, vol. 11, no. 2, pp. 11–17, 2008.

[10] S. V. Moreno-Tapia, L. A. Vera-Salas, R. A. Osornio-Rios, A. Dominguez-Gonzalez, I. Stiharu, and R. D. J. Romero-Troncoso, "A field programmable gate array-based reconfigurable smart-sensor network for wireless monitoring of new generation computer numerically controlled machines," *Sensors*, vol. 10, no. 8, pp. 7263–7286, 2010.

[11] L. Ruiz-Garcia, P. Barreiro, J. I. Robla, and L. Lunadei, "Testing zigBee motes for monitoring refrigerated vegetable transportation under real conditions," *Sensors*, vol. 10, no. 5, pp. 4968–4982, 2010.

[12] Z. Rasin and M. R. Abdullah, "Water quality monitoring system using zigbee based wireless sensor network," *International Journal of Engineering & Technology*, vol. 9, no. 10, pp. 24–28, 2009.

[13] "Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (LRWPANs)," IEEE Standard 802.15.4, 2006.

[14] L. Ruiz-Garcia, L. Lunadei, P. Barreiro, and J. I. Robla, "A review of wireless sensor technologies and applications in agriculture and food industry: state of the art and current trends," *Sensors (Switzerland)*, vol. 9, no. 6, pp. 4728–4750, 2009.

[15] R. Garcia, A. Gordon-Ross, and A. D. George, "Exploiting partially reconfigurable FPGAs for situation-based reconfiguration in wireless sensor networks," in *Proceedings of the 17th IEEE Symposium on Field Programmable Custom Computing Machines (FCCM '09)*, pp. 243–246, Napa, Calif, USA, April 2009.

[16] B. O'Flynn, S. Bellis, K. Delaney et al., "The development of a novel minaturized modular platform for wireless sensor networks," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*, pp. 370–375, Boise, Idaho, USA, April 2005.

[17] O. Postolache, P. S. Girão, J. M. D. Pereira, and H. Ramos, "Water quality sensors calibration system based on reconfigurable fpga technology," in *Proceedings of the 18th IMEKO World Congress*, September 2006.

[18] S. Rana and R. Gill, "Sensor fpga bas ed remote monitoring system for food preservation," *International Journal of Innovative Technology and Exploring Engineering*, vol. 1, no. 2, 2006.

[19] S. Saponara, E. Petri, L. Fanucci, and P. Terreni, "Smart transducer interface in embedded systems for networked sensors based on the emerging IEEE 1451 standard: H2 detection case study," in *Proceedings of the 7th Workshop on Intelligent Solutions in Embedded Systems (WISES '09)*, pp. 49–55, June 2009.

[20] J.-F. Tu, "Utilizing uart method for realizing the tii of ieee 1451," *International Journal of Electronics Engineering*, vol. 1, no. 2, pp. 133–139, 2009.

[21] P. P. Chu, *FPGA Prototyping by VH DL Examples, Xilinx Spartan-3 Version*, John Wiley & Sons, 2008.

[22] R. Gallo, M. Delvai, W. Elmenreich, and A. Steininger, "Revision and verification of an enhanced UART," in *Proceedings of the IEEE International Workshop on Factory Communication Systems (WFCS '04)*, pp. 315–318, September 2004.

[23] M. Kaur and R. Mittal, "Fpga implimentation & design of microuart with different baud rates," *Journal of Information Systems and Communication*, vol. 3, no. 1, pp. 41–44, 2012.

[24] J. Kamala and B. Umamaheswari, "Ieee 1451.0-2007 compatible smart sensor readout with error compensation using fpga," *Sensors & Transducers Journal*, vol. 102, no. 3, pp. 10–21, 2009.

[25] *DS18B20 Programmable Resolution 1-Wire Digital Thermometer Data Sheet*, Maxim Integrated Products, San Jose, Calif, USA, 2008.

[26] UNEP GEMS/Water Programme of IAEA, *Analytical Methods for Environmental Water Quality*, United Nations Environment Programme Global Environment Monitoring System (GEMS)/Water Programme in International Atomic Energy Agency, 2004.

[27] "Xilinx power estimator," http://www.xilinx.com/products/design_tools/logic_design/xpe.htm.

[28] "Fpga xilinx xc3s250e mcu core board core3s250e," http://www.wvshare.com/product/Core3S250E.htm.