DEVELOPING A SENSOR FIRMWARE APPLICATION FOR REAL-LIFE USAGE

Hristijan Gjoreski, Mitja Luštrek, Matjaž Gams
Department of Intelligent Systems, Jožef Stefan Institute,
Jožef Stefan International Postgraduate School,
e-mail: {hristijan.gjoreski, mitja.lustrek, matjaz.gams}@ijs.si

ABSTRACT

In recent years the demand for intelligent systems that support the life of the elderly is increasing. In order to provide an appropriate support, these systems should constantly monitor the user with sensors. However, using sensors in real-life situations is a challenging task, mainly because of the constraints in the sensor energy consumption (battery life) and memory capacity for storing the sensors data. In this paper we present an example of a sensor communication protocol developed for the Shimmer accelerometers, so that they can be used in in real-life situations, i.e., constantly monitoring the user during a normal day. A custom firmware application is developed, which has several functionalities: real-time data streaming through Bluetooth, data logging into internal microSD card, sending the stored data to a Bluetooth-enabled device and detecting when the sensor is put on and off a charging dock.

1 INTRODUCTION

The world's population is aging rapidly, threatening to overwhelm the society's capacity to take care of its elderly members. The percentage of persons aged 65 or over in developed countries is projected to rise from 7.5% in 2009 to 16% in 2050 [1]. This is driving the development of innovative ambient assisted living (AAL) technologies to help the elderly live independently for longer and with minimal support from the working-age population [2][3]. To provide timely and appropriate assistance, AAL systems must monitor the user by using ambient (environmental) and/or wearable sensors. With the recent development of the sensors technology, the wearable sensors are gaining attraction and can measure lot of different user-related parameters: location, activity, physiological, etc. Examples of them include: GPS, accelerometers, gyroscopes, heart-rate sensors, breath-rate sensors, etc.

In order to be used in everyday life situations, these sensors have to able to constantly monitor the user during the day. However, often this is a challenging task mostly due to battery consumption constraints and memory storage capacity. In this study we present a sensing protocol for the Shimmer accelerometer sensors, so they can be used to constantly monitor the user during the day. A custom firmware TinyOS application was developed in order to satisfy the user's requirements and the sensors limitations. It implements two modes of operation: real-time data sending and logging the data in the internal memory and sending it offline in batches.

The motivation and the context of the study is the CHIRON project (Cyclic and person-centric health management: Integrated approach for home, mobile and clinical environments) [4]. It is a European research project of the ARTEMIS JU Program with 27 project partners. It includes industry partners (large companies and SMEs), research and the academic institutions, and also medical institutions. The project addresses one of the today's societal challenges i.e., "effective and affordable healthcare and wellbeing". CHIRON combines state-of-the art technologies and innovative solutions into an integrated framework of embedded systems for effective and person-centered health management throughout the complete healthcare cycle, from primary prevention (person still healthy) to secondary prevention (initial symptoms or discomfort) and tertiary prevention (disease diagnosis, treatment and rehabilitation) in various domains: at home, in nomadic environments and in clinical settings.

2 SENSORS

In the CHIRON project, two Shimmer accelerometers are used to monitor the user's activities. The sensor platform is based on the Shimmer Wireless Sensor Network (WSN) module. It is based on a T.I. MSP430F1611 microcontroller, which operates at a maximum frequency of 8 MHz and is equipped with 10Kb RAM and 48 Kb of Flash. Wireless communication is achieved either with Bluetooth v2 (BT – RN-42 module) or through IEEE 802.15.4 (T.I. CC2420 module.). In our study we used the standard BT v2 in order to easily connect it with a smartphone. For storage purposes, the Shimmer platform is equipped with an integrated 2GB microSD card, which is used in normal operation mode to store sensor readings [5]. The power supply is comprised of a 450mAh rechargeable Li-ion battery.

The firmware of the Shimmer platform is based on the open-source TinyOS operating system [6]. It uses the NesC programming language, which is a light-weight version of C. TinyOS/NesC is dedicated for low-power wireless

sensors and allows many sensor platforms with a heterogeneous set of hardware devices to be programmed and controlled (microcontroller, sensors, SD cards, etc.).

The TinyOS in the Shimmer sensors follows three-layer abstraction architecture. At the bottom is the Hardware Presentation Layer (HPL) which allows access to input/output pins or registers of the hardware devices. Next, the Hardware Abstraction Layer (HAL) allows configuring more complex functionality in order to communicate with external sensors or resources implemented in the platform. The top layer is the Hardware Independent Layer (HIL), which permits to read the sensor data independently of the digital communication bus.

Each layer communicates with the adjacent ones trough interfaces, either generic or hardware specific. As the TinyOS is an event-driven operating system, the interface call commands that are addressed to the lower layer. These commands are answered from the lower layers by signaling events. In our case, the Shimmer sensor platform, the HPL and the HAL layers are already available and for its internal resources, as the accelerometer, the SD card or the Bluetooth radio, the HIL layer is also implemented.

Once the layers are implemented, a firmware application is developed. In our case, the application is based on the specifications (sensing protocol) provided by doctors in the CHIRON project. The firmware application and the sensing protocol are discussed in the next section.

3 SENSING PROTOCOL AND FIRMWARE APPLICATION

In order to explain the sensing protocol (shown in Figure 1), let us consider the following scenario. The user wakes up and takes the two accelerometers from the charging dock. Once they are taken out from the dock, the sensors have to start sensing. The user attaches the sensors in the wearable garment (e.g., chest and thigh elastic straps) and performs an initialization activity sequence. This sequence is performed in order to ensure if the sensors have the right orientation (important for the post-processing of the data). The orientation checking lasts for a few minutes, during which the sensor data is streamed in real-time to a smartphone application. Once the smartphone confirms that the setup is all right, the user continues with his everyday activities. During this period the sensors log the data locally to a microSD card. At the end of the day, the user takes out the sensors, puts them to the charging docks, and goes to sleep. During the sleep, the sensors are charged and all the data is transferred to the processing unit.

The scenario shows that the battery life of the sensors should last at least 16 hours (the active period of a normal day) and the sensors should be able to receive commands from a smartphone through Bluetooth. Our tests showed that if the standard firmware application is used (real-time data sending using BT), the battery will last around 6 hours, which is not sufficient for the whole day. Furthermore, with this approach there should be a constant BT connection between the smartphone and the sensors, which is highly unlikely in real-life scenario. In order to achieve these functionalities we created a custom firmware application which has two modes of operations: real-time data sending and logging the data in microSD card and sending it for offline analysis. The application is based on the two standard Shimmer firmware applications, which are publicly available: real-time data acquisition ("BoilerPlate.ihex") and data logging ("JustFATLogging.ihex").

The original logging application (JustFATLogging.ihex) has one main function, to log the acceleration data on the microSD card. The start of the logging is triggered when the sensor is removed from a dock station and the end of logging is triggered when the sensor is put back on the dock station. The data can be accessed only through the USB port of the docking station. We used this firmware application as a base for further development.

First, we added the Bluetooth functionality in order to allow wireless communication between the sensor and the smartphone. However, the activation of the Bluetooth significantly decreased the sensor's battery life. Therefore, we modified the application so the Bluetooth is activated only when the sensor is put back on the charger. During the charging time, the smartphone sends a command and collects the logged data. When the user decides to mount the sensors he/she gives a command to start logging and to turns off the Bluetooth. Thus, during the logging process the Bluetooth is off and there is no communication between the smartphone and the sensor.

For the acceleration data, it is really important how the sensor is mounted, i.e., the sensor orientation must be the same for every recording. In order to check the orientation of the sensors, an algorithm analyzes the data during some predefined activities, e.g., standing and lying, and accordingly gives a notification to the user if the sensors are mounted in the correct way. To allow this data analysis, the data has to be analyzed in real-time, therefore we added also this functionality, i.e., *real-time transmission*. The real-time data acquisition is performed before the *start logging* command is sent.

In addition, two more functionalities were implemented: deleting a log file (*delete log*), and checking the availability of a log file (*is log available*).

The final modification is related to the timestamps of the data samples and data synchronization between different sensors. In order to synchronize the data between the sensors, one must know the absolute timestamp of the data samples. In our case, we used the timestamp of the start of the logging and the time difference between consecutive data samples. The sensor's internal crystal clock is used for estimating the time difference between consecutive data samples. Thus, each data sample is labeled with a timestamp provided by the clock. The starting timestamp is sent by the smartphone with each start logging command. Using the starting timestamp and the internal counter's timestamps, the smartphone was able to reconstruct an absolute timestamp for each data sample.

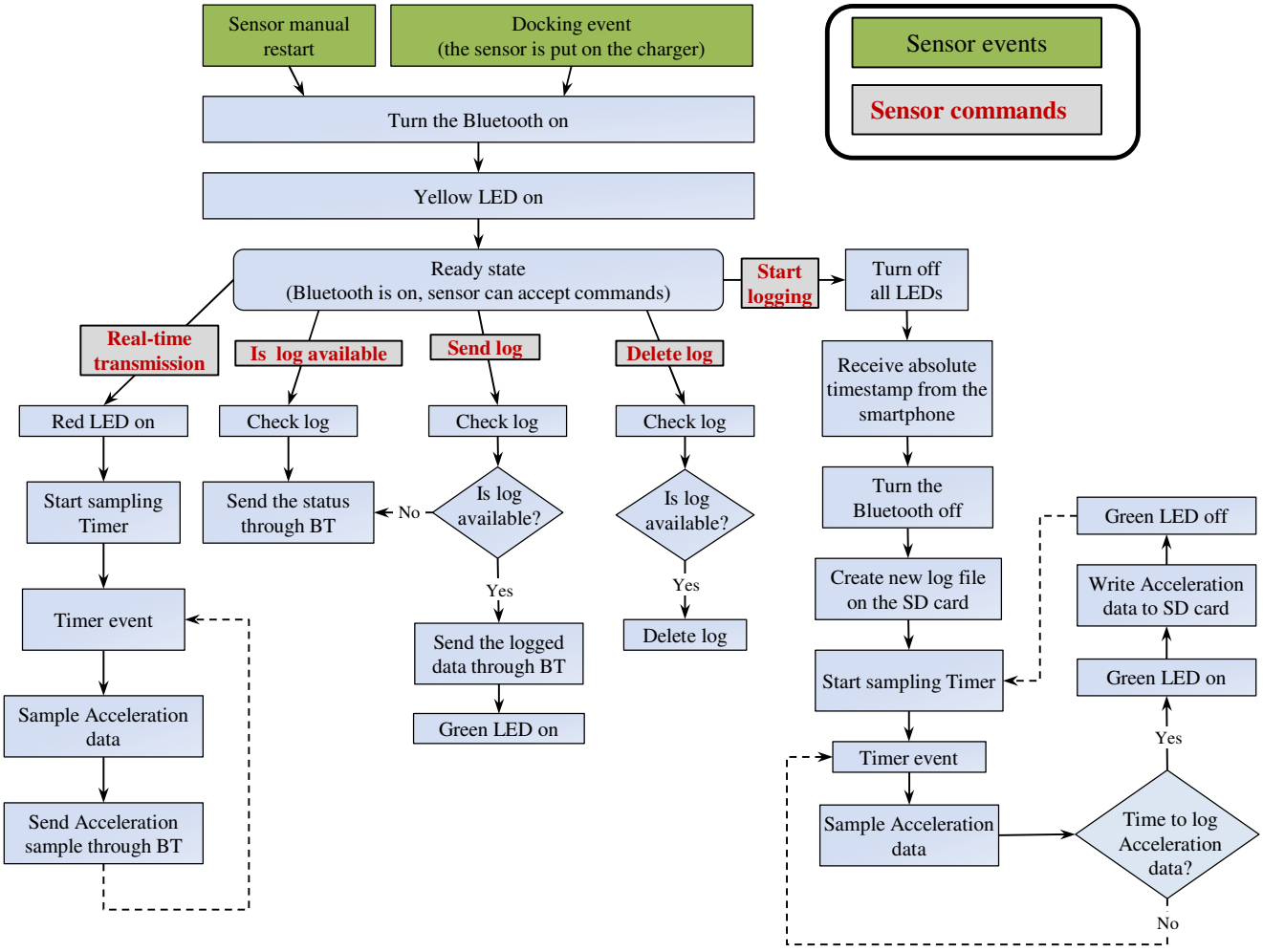

Figure 1. Sensor Firmware Flowchart.

In theory, the internal Shimmer crystal clock (Epson FC-135 32.7680KA-A3) tolerance is ±20 ppm, which results in 1.8 seconds maximal drift in 24 hours. In the worst case scenario, when two sensors have different drift direction (+ or -), the time difference is 3.6 seconds, which is acceptable for the project's requirements. Several practical tests were performed and confirmed the theoretical analysis, i.e., the measured drift was in the range of 1 second for a whole day recording.

4 SENSOR COMMANDS AND EVENTS

Table 1 describes the commands that can be received by the sensor application firmware. These commands can be sent by any BT-equipped device.

Table 1. Sensor commands.

Real-time transmission	The sensor samples accelerometer data at 50Hz and sends each sample to the smartphone.
Start logging	The sensor stops the Real-time sending, and waits for the absolute real timestamp from the smartphone. This timestamp is written at the beginning of each log file and is used as a reference point for reconstructing the timestamp for each data sample. After that, the sensor starts logging the accelerometer data. Additionally, the Bluetooth is turned off; there is no communication between the sensor and the smartphone during the logging process.
Is log available	The sensors checks if the log file is available for sending and sends the status.
Send log	The sensor sends the logged file. First, the absolute timestamp is sent, and then the accelerometer data samples are sent.
Delete log	The sensor deletes the log file.

Table 2 describes the events that are detected by the application firmware.

Table 2. Events that can be detected by the sensor's application firmware

Docking event (the sensor is put on the charging dock)	The sensor stops the logging and turns on the Bluetooth. Yellow LED is turned on, representing that the log file is ready to be sent.
Sensor manual restart	The sensor restarts to the initial state. That is, stops the logging and turns on the Bluetooth. Yellow LED is turned on, representing that the log file is ready to be sent.

5 THEORETICAL AND EMPIRICAL TESTS

After developing the firmware application we performed several theoretical and empirical tests. First, we analyzed the amount of data expected to be generated on a daily basis. The MSP430 A/D channels perform 12-bit (2 bytes of storage) digitization and that a 16 bit (2 bytes) timestamp is stored for each sample. Table 2 summarizes our projections based on the sampling frequency of every sensor. Based on these calculations the total amount of data for 12-16h of daily use should be 114Mb – 152.4Mb. This amount of data does not pose any issue in any operational mode, since in the real-time scenario BT can achieve data rates up to 300kbps which is more than adequate for the amount of data generated per second and in the logging operating mode the microSD cards on the modules have more than enough capacity to store the generated data.

Table 3. The amount of data generated by the accelerometer.

Sampling Frequency (Hz)	Data per second (KB/s)	Data per hour (MB/h)
50	0.78	2.74

The energy consumption analysis of the Shimmer platform, presented by Burns et. al. [5], designates that the accelerometer draws 1.6mA when sampled at 50Hz. When the sensor streams accelerometer data in real-time through the BT the consumption increases to 5.2 mA. From this analysis, it is safe to assume, that since the same hardware equipped with a 450mAh battery is used the clinical requirement of 6-8h data logging (in the storing mode) or an adequate amount of time (around 1h) for live streaming (streaming mode) can be easily met, provided that the module's batteries are fully charged at the beginning of sensing. Table 4 lists the average battery lifetime (full battery drainage period) obtained for the two modes of operation from a series of experiments.

Table 4. Average working time for the real-time (Bluetooth is active) and the logging mode (Bluetooth is not active; the sensor is logging in the SD card).

Sampling Frequency (Hz)	Real-time mode	SD-logging mode
50	6h 30m	14 days

6 CONCLUSION

In this paper we showed how one can overcome the sensor limitations (battery life and memory storage) by creating a custom firmware application and adjusting it to real-life situations. We presented a sensing protocol and sensor firmware application developed for the Shimmer accelerometers. The protocol was created so that the sensors can be used in in real-life situations, i.e., constantly monitoring the user during a normal day. The developed custom firmware application has several functionalities: real-time data streaming through Bluetooth, data logging into internal microSD card, sending the stored data to a Bluetooth-enabled device, and detecting when the sensor is put on and off a charging dock.

Acknowledgement

This work was partly supported by the Slovene Human Resources Development and Scholarship funds and partly by the CHIRON project - ARTEMIS Joint Undertaking, under grant agreement No. 2009-1-100228.

References

[1] United Nations 2009, World population ageing, Report

[2] A. Bourouis, M. Feham, A. Bouchachia, "A new architecture of a ubiquitous health monitoring system: a prototype of cloud mobile health monitoring system," The Computing Research Repository, 2012.

[3] M. Luštrek, B. Kaluža, B. Cvetković, E. Dovgan, H. Gjoreski, V. Mirchevska, M. Gams, "Confidence: ubiquitous care system to support independent living" DEMO at European Conference on Artificial Intelligence, pp. 1013-1014, 2012.

[4] The CHIRON project: http://www.chiron-project.eu/

[5] A. Burns, B. R. Greene, M. J. McGrath, T. J. O'Shea, B. Kuris, S. M. Ayer, F. Stroiescu, V. Cionca, "SHIMMER™ – A Wireless Sensor Platform for Noninvasive Biomedical Research," IEEE Sens. J., vol.10, no.9, pp.1527- 1534, 2010.

[6] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, D. Culler. TinyOS: An operating system for sensor networks. In Werner Weber, JanM Rabaey, and Emile Aarts, editors, Ambient Intelligence, chapter 7, pp. 115–148, 2005.