Conference on Electronics, Telecommunications and Computers – CETC 2013

Performance Comparison of Low-cost Hardware Platforms Targeting Physiological Computing Applications

José Guerreiro\textsuperscript{a,b}, André Lourenço\textsuperscript{a,b}, Hugo Silva\textsuperscript{b}, Ana Fred\textsuperscript{b}

\textsuperscript{a}Instituto Superior de Engenharia de Lisboa, Rua Conselheiro Emídio Navarro, 1, 1959-007 Lisboa, Portugal
\textsuperscript{b}Instituto de Telecomunicações, Instituto Superior Técnico, Avenida Rovisco Pais, 1, 1049-001 Lisboa, Portugal

Abstract

Although the Arduino is currently the most popular low-cost hardware platform within the Do-it-Yourself (DiY) community, it is not designed taking into account the specific requirements of physiological computing. As a result of previous work by our group, a novel hardware framework for physiological computing, entitled BITalino, was proposed, to allow anyone to create their own projects and applications with biosignal sensors. In this work, we benchmark the Arduino and BITalino platforms, targeting the evaluation of their performance for the purpose of biosignals acquisition. We also describe a few physiological computing applications built with the BITalino, that show the potential of this platform within the engineering community.

© 2014 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/3.0/).
Peer-review under responsibility of ISEL – Instituto Superior de Engenharia de Lisboa, Lisbon, PORTUGAL.

Keywords: Biosignals, Physiological Computing, Do-it-Yourself (DiY), Arduino, BITalino.

1. Introduction

The most recent trends in low-cost and open-source hardware are playing a transforming role on several engineering disciplines, in areas including computer science, informatics, electrical engineering, among others. Furthermore, evidence shows that the Do-it-Yourself (DiY) hardware movement is clearly a topical field that is boosting the creativity within the engineering community.

On the other hand, biosignals have become an increasingly important topic of interest in applications that go beyond the classical medical domain [1]. The modern uses of biosignals include human enhancement during sports activity [2], human-computer interaction on computer gaming [3,4], biofeedback [5], and biometric [6,7] applications, which are generating a growing interest within the global research and industrial engineering landscape.

The most broadly known DiY electronics prototyping platform is the Arduino [8], which provides an easy-to-use hardware and software environment accessible for anyone, and is showing a vibrant community of variants and followers [9,10]. While the Arduino is often employed in projects within physical computing and robotics, the physiological computing community has been mostly lacking a comparable tool. In previous research from our group, we built on...
the guiding principles of the low-cost DiY hardware community and proposed BITalino, a toolkit designed to allow anyone to create their own applications with biosignal sensors [11,12]. It is low-cost, highly versatile and by analogy to the concept of physical computing, it is an enabling technology for physiological computing.

In this paper we benchmark the differences between Arduino and BITalino, making a comparative assessment targeting biosignal acquisition. We also build on our previous work, by presenting a few application examples in which BITalino is used as a novel hardware framework for physiological computing.

The rest of the paper is organized as follows: Section 2 provides a brief overview of the Arduino and BITalino platforms; Section 3 describes the tests used to compare the two platforms, namely, quantitative tests to characterize temporal uncertainty in terms of sampling rate accuracy and dynamic specifications of the Analog-to-Digital Conversion (ADC) process; Section 4 presents application examples where the BITalino has shown to be particularly relevant, demonstrating its potential for the physiological computing community; and finally, Section 5 outlines the main conclusions and future work.

2. Overview

This Section summarizes the main specifications of the Arduino and the BITalino platforms. Figure 1 illustrates the overall design and form factor of each platform, while the main specifications that an average user has available are summarized in Table 1.

| Feature            | BITalino          | Arduino Uno | Arduino Pro Mini |
|--------------------|-------------------|-------------|------------------|
| MCU                | ATMEGA328P        | ATMEGA328P  | ATMEGA328P       |
| Clock [MHz]        | 8                 | 16          | 8/16             |
| Power [V]          | 3.3               | 5           | 3.3/5            |
| Data Link          | Bluetooth (default) + UART | USB (default) + UART + I2C + SPI | UART + I2C + SPI |
| On-board Sensors   | ECG/EMG/EDA/ACC/LUX | -           | -                |
| On-board Actuators | LED               | LED         | LED              |
| Dimensions [cm]    | 6.0 × 10.5        | 5.4 × 6.8   | 1.8 × 3.4        |

2.1. Arduino

The basic hardware consists of a Microcontroller Unit (MCU) based on an 8-bit Atmel AVR Reduced Instruction Set Computer (RISC) microcontroller (e.g. Arduino Pro Mini and Uno versions) running at a clock speed up to 8/16MHz, with 32kB flash memory (0.5kB are used by bootloader) and 2kB Static Random Access Memory (SRAM). In addition, it has included on-board, as standard features, 11 digital pins which can be set as either an input or output as well as 7 analog input pins (e.g. Arduino Pro Mini). It is also designed with standard pinout shields, which
provide expansion for its basic functionalities. Another important feature of the Arduino platform is the embedded bootloader that executes on the microcontroller, allowing it to be reprogrammed via a USB-to-serial converter. The standard boards have a voltage regulator, and can be powered either using an USB or DC source. For programming, the Arduino software IDE (working environment) was designed to make it easier to develop embedded projects. It consists of a standard JAVA-like programming language compiler, and it also includes basic libraries to control or communicate with common devices such as LEDs, LCDs, among other peripherals.

2.2. BITalino

BITalino is an all-in-one board that was designed as a set of modular blocks to allow maximum versatility. It integrates multiple measurement sensors on-board for bioelectrical and biomechanical data acquisition. Furthermore, it includes a digital back-end supported by an 8-bit Atmel AVR-RISC microcontroller (ATMEGA328P) which runs at 8 MHz CPU clock speed, and has 2kB SRAM and 32kB flash memory, a power management block (for circuit voltage regulation and battery charging), and a Bluetooth wireless communication module for data transfer to a base station (e.g. laptop, tablet, mobile phone, etc).

BITalino also enables each individual block to be detached from the main board, and integrates two auxiliary connectivity blocks on the board, which enable RJ22 plugs to be added. This configuration allows the users to implement customized sensor configurations, or even to add other sensors connected with cables. BITalino has a purpose-built firmware for physiological signal acquisition, which executes on the microcontroller and manages the behaviour of the system, controls the data streaming over the UART bus, and enables the configuration of multiple acquisition parameters. We refer the reader to [11,12] for overall characterization and additional information about BITalino.

Table 2. Results for the temporal uncertainty comparison between Arduino and BITalino, $Fs = 1kHz$ sampling rate

| Platform | # channels | firmware            | $Fs$ (real value)[Hz] | Skew[ %] | Jitter[ %] |
|----------|------------|---------------------|-----------------------|----------|------------|
| BITalino | 1          | BITalino firmware   | 999.995 ± 0.025       | 0.0005   | 0.0025     |
| Arduino Pro Mini | 1 | BITalino firmware | 1002.210 ± 0.044 | 0.2210 | 0.0044     |
| Arduino Pro Uno | 1 | BITalino firmware | 1001.905 ± 0.015 | 0.1905 | 0.0015     |
| Arduino Pro Mini | 1 | Arduino reference library | 997.832 ± 0.261 | 0.2167 | 0.0261     |
| Arduino Pro Uno | 1 | Arduino reference library | 1004.931 ± 0.391 | 0.4931 | 0.0389     |
| BITalino | 6          | BITalino firmware   | 999.9914 ± 0.025     | 0.0009   | 0.0025     |
| Arduino Pro Mini | 6 | BITalino firmware | 1002.320 ± 0.047 | 0.2315 | 0.0047     |
| Arduino Pro Uno | 6 | BITalino firmware | 1001.809 ± 0.016 | 0.1809 | 0.0016     |
| Arduino Pro Mini | 6 | Arduino reference library | 984.472 ± 0.456 | 1.5528 | 0.0463     |
| Arduino Pro Uno | 6 | Arduino reference library | 990.192 ± 0.234 | 0.9808 | 0.0237     |

3. Comparative Tests

 Biosignals have very strict requirements in terms of signal acquisition parameters. Consequently, biosignal acquisition instrumentation has to be designed with high noise rejection capabilities and accurate sampling rates; these being fundamental requirements.

Tests were performed to compare the BITalino with the Arduino Pro Mini (3.3V - 8MHz) and the Arduino Uno (5V - 16MHz) platforms, namely, the temporal uncertainty (i.e. sampling rate accuracy) and the dynamic specifications of the ADC process, in particular: Crosstalk between adjacent channels; SNR: Signal-to-Noise Ratio; THD: Total Harmonic Distortion; SINAD: Signal-to-Noise Ratio plus Distortion and ENOB: Effective Number of Bits. For these experimental tests, synthesized signals were generated using an Agilent 33220A Function Waveform Generator.

To characterize the temporal uncertainty of the platforms, we acquired a synthesized ramp wave with $1kHz$ frequency, $3V_{pp}$ dynamic range, and $V_{cc}/2$ offset. We then analysed its slope ($\frac{\Delta V}{\Delta t}$) and compared with the slope of the
Table 3. Comparative test of the dynamic specification of the ADC between Arduino boards (both gave identical results - Mini/Uno) and BITalino, (15Hz sine wave; $F_s = 1kHz$)

| Parameters       | Arduino boards | BITalino    |
|------------------|----------------|-------------|
| Crosstalk[dB]    | -0.43          | -61.96      |
| SNR[dB]          | 23.97          | 55.72       |
| THD[dBc]         | -49.94         | -59.80      |
| SINAD[dBc]       | 23.95          | 54.29       |
| ENOB[bits]       | 3.69           | 8.73        |

signal that was really injected. Table 2 shows the sampling rate accuracy results using a sampling rate of 1kHz in all devices; it shows that there is higher sampling rate drift and skew in both Arduino models when compared with the BITalino. From the experimental results, the main reason for this being the fact that BITalino has an external Quartz Crystal oscillator as clock source, with 8MHz frequency and ±20ppm of frequency stability and tolerance. Together with the purpose-built firmware that has no bootloader as well as no Arduino library overhead, BITalino ensures a high accuracy conversion of the analog signals to a digital format.

On the other hand, the results show that when the Arduino microcontrollers are programmed using its reference library [13], the sampling rate drift is high and also increases according to the number of channels acquired. When the firmware that was executed on the Arduino microcontrollers was the purpose-built BITalino firmware, the jitter percentage slightly decreased but the sampling rate drift is still high. The reason for this is the fact that the MCU on most Arduino boards is clocked by a less accurate clock source (with error of approximately 0.2%), which could not be adequate for such applications involving physiological data that would require a precise system which also preserves the signal morphology, in particular those more directly related with health or/biometry. Other applications on the other hand, may not require such precise acquisition when just involve controlling a device using a binary threshold according to some physiological response, as Electromyographic (EMG) signals, namely, to trigger a door lock or to switch on/off a light.

Fig. 2. 15Hz sine wave acquired at 1kHz sampling rate, using the Arduino platforms and BITalino. The first row corresponds to Arduino results (both gave identical results - Mini/Uno), while the second to BITalino results; the first column (a1, b1) illustrates the original signal acquired with both platforms, respectively; the second column (a2, b2) illustrates the noise present in the signal excluding DC, the original signal itself and harmonics; the third column (a3, b3) shows the frequency response of the noise spectral components excluding DC, the original signal and its harmonics; and the fourth column (a4, b4) shows the noise amplitude histograms.
Table 3 shows the dynamic specification results for the ADC performed with the Arduino boards and BITalino. We computed the SNR, THD, SINAD and ENOB, which are typical parameters used for evaluating the ADC performance [14]. During this test a synthesized sine wave was injected, with 15Hz of frequency and voltage range about 95% of the full-scale range of the ADC. The results show that the analog-to-digital conversion process within the Arduino platforms are more sensitive to noise; they also have higher harmonic distortion (mainly driven by the inaccuracy of the sampling rates) and crosstalk between channels when compared with the BITalino. In addition, they have a very low ENOB, which means that the discretization performed by the analog-to-digital conversion is less accurate.

Figure 2 illustrates the synthetic signals used to compute the results presented in Table 3, and auxiliary signals used to support our conclusions. Given the sampling rate skewness on the Arduino boards, some additional spectral components around the fundamental harmonic can be found in Figure 2 (a2, a3). This issue increases the THD and also degrades the SNR.

In what concerns the noise distribution, as illustrated by the histograms (Figure 2 a4, b4), both Arduino and BITalino present a Gaussian distribution, as supported by the Kolmogorov-Smirnov test (p-values < 0.05 in both situations) [15]. It is important to notice that the standard deviation of the inferred gaussians is $\sigma = 21.9$ and $\sigma = 0.5$, for the Arduino (both Mini/Uno) and BITalino, respectively. We refer the reader to [11,12,16] for additional performance characterization of the BITalino.

4. BITalino Application Examples

In this Section we present examples of applications built using the BITalino platform, and which demonstrate its potential, namely, a heartbeat detector that uses Electrocardiographic (ECG) signals to trigger a LED (Beat-by-BIT); a gesture-controlled light based on Accelerometer (ACC) signals (LightBIT); a door lock that uses Electromyographic (EMG) signals as a trigger (LockBIT); and a didactic and interactive setup that responds to the emotional arousal variations as determined from Electrodermal Activity (EDA) and Heart Rate (HR) ("Mentir de Verdade").

4.1. Beat-by-BIT

The Beat-by-BIT is a heartbeat detection application, that uses ECG signals to trigger the onboard BITalino LED. This application uses a BITalino Freestyle [12], with ECG sensor, MCU block, Power block and Battery (Figure 3a). Using standalone firmware, this system analyses the ECG signal to detect the R peaks within the P-QRS-T-U complexes (Figure 3b).

![BITalino Freestyle device and ECG signal waveform](image)

(a) BITalino Freestyle device. (b) Prototypical ECG waveform.

Fig. 3. Beat-by-BIT device and ECG signal waveform.

The strategy used for peak detection is adapted from the Slope Sum Function (SSF) algorithm to detect the onset of the arterial blood pressure pulses as proposed in [17]. As shown in Figure 4, the algorithm consists of three stages: a low-pass filter, a slope sum function, and a decision rule. The ECG signal, $x_n$, is the input of the low-pass filter, and $y_n$ is the filtered ECG signal. The slope sum function converts $y_n$ to a slope sum signal $z_n$. A decision rule is applied to $z_n$ to determine the R peak instants, $t$, in time.
The purpose of the low-pass filter is to suppress high frequency noise that might affect the R peak detection, smoothing the ECG signal. We used a moving average filter, as described by Equation 1, where $x_n$ is the input signal, $y_n$ is the output signal, and $M$ is the moving average window size. For a sampling frequency of 1000 Hz, the $-3dB$ cut-off frequency is about 30Hz and the gain is 20 at 0 Hz. The phase shift is 20ms (20 samples).

$$y[n] = \frac{1}{M} \sum_{j=0}^{M-1} x[n + j]$$ (1)

The purpose of the SSF is to enhance the slope introduced by the rapidly changing R peak within the QRS complex, and to suppress the remainder of the ECG signal. The slope sum function, $z_n$, at time $n$, is defined in Equation 2.

$$z_n = \sum_{k=n-j}^{n} \Delta u_k, \Delta u_k = \begin{cases} \Delta y_k > 0 \\ 0 \quad \Delta y_k \leq 0 \end{cases}$$ (2)

Where $\Delta y_k = y_k - y_{k-2}$ is the 2nd order discrete derivative that results in a band-pass filter. The onset of the SSF signal coincides with the onset of the R peak, given that the SSF signal rises when the ECG signal rises.

Finally, a decision rule that allows the detection of each SSF pulse onset is applied. First, we defined an amplitude interval by establishing empirical thresholds ($thd_{min} = 150$ and $thd_{max} = 400$), based on experimental studies and adapted from [18] for this application. The algorithm rejects the values within the SSF signal that fall outside of the threshold interval, and to avoid double detection of the same pulse, the algorithm only accepts heartbeats between 40bpm and 180bpm. All newly accepted SSF pulses are evaluated according to this "heartbeat" rule. Figure 5 illustrates all the signals resulting from each processing stage previously described. The power consumption of this purpose-built device is about $\approx 50mA$, which corresponds at 0.185W. Tests with a Lithium-Ion/Lithium-Polymer battery with 3.7 V - 850mA h have shown that the system has an autonomy of $\approx 17$ hours when in continuous real-time operation.
4.2. Other Examples

The LockBIT is a muscle-controlled door lock application, that uses EMG signals as a trigger; in Figure 6a we show the EMG sensor placement. The LightBIT is a gesture-controlled light application, that is based on ACC signals to switch on/off a light when the BITalino board is moved.

These applications use two BITalino versions: a BITalino Board to acquire the EMG or ACC signals and send the data to a base station through the Bluetooth module, and a BITalino Plugged [12] to trigger the door lock or the light circuits. A basic onset and motion detection software in Python running in the base station, allows real-time data processing from each BITalino. The data from the BITalino Board is analysed using a moving average window of 100 samples and validated against a threshold value previously defined. If the thresholding criteria is met, the base station sends a command to the BITalino Plugged (as illustrated in Figure 6b for the door lock example) to trigger one of the digital output ports, which is directly connected to the door lock or the light circuits via a switching circuit illustrated in Figure 6c.

Another example is a classical and well known use of electrophysiological signals within the polygraph test, usually known as a "lie-detector". Typically, the instrument used to conduct polygraph tests consists of a physiological recorder that assesses indicators of autonomic arousal [20], such as: heart rate, blood pressure, respiration and electrodermal activity. Emotional situations (i.e. stress) generate variations on the perspiration of the skin and variations of the heart rate as well. Those can be obtained through EDA signals and the Blood Volume Pulse (BVP) signals, respectively, since it is directly related to emotional variations, and as such, can be related with a lie.

In the "Mentir de Verdade" project we addressed this issue in a pedagogical way, building a demonstration system where two participants are involved in a role-playing game based on the lying detection. The instrumentation setup is taken non-intrusively and in an off-the-person approach. The system is focused on detecting physiological differences triggered by deceptive answers, based on the EDA signals measured by a BITalino EDA sensor and the BVP signals measured by an off-the-shelf sensor [21], both attached to the hand, and using the BITalino as the platform for signals acquisition. Figure 6d depicts an acquisition record of the obtained BVP and EDA signals, and also the HR variation over time. Each red timestamp indicates the instant when the game starts, while the grey region indicates the interval where the lie occurred. We refer the reader to [19] and [22] for additional characterization of this interactive "lie-detector" game.

5. Conclusion

In this work we have presented a detailed comparison between two DiY hardware platforms, namely the Arduino and the BITalino. Biosignal acquisition requires adequate instrumentation, designed with strict requirements in terms of tolerance to noise and sampling rate accuracy, such as BITalino for a broad range physiological computing applications with different demands. However, together with suitable off-the-shelf physiological sensors, the Arduino platforms could also be included in many applications.
BITalino is a board for the DiY community, but its specifications allow a rigorous treatment in regarding to biosignal acquisition. Although the data collected through both platforms can preserve the waveform properties, experimental results have shown that BITalino is several orders of magnitude more accurate than the Arduino, with the additional advantage of being more robust to noise. Additionally, we described several applications that demonstrate the potential of BITalino, namely: a heartbeat detector based on ECG to trigger a LED; a muscle-controlled door lock, that uses EMG signals as a trigger; a light controlled by gestures, using ACC signals; and a didactic and interactive setup for detection of deception that answers according to the emotional variations based on EDA and HR.

Future work will be mainly focused on a slight review of some components that integrate the BITalino board in order to lower its price and provide an easier manufacturing process. Additionally, new modules will be designed, namely, analog sensors, and a SD Card module for standalone signal acquisition. We also aim to improve the communication module for a Bluetooth Low Energy (BLE), to ensure less power consumption.

6. Acknowledgements

This work was partially funded by Fundação para a Ciência e Tecnologia (FCT) under the grants PTDC/EEI-SII/2312/2012, SFRH/BD/65248/2009 and SFRH/PROTEC/49512/2009, whose support the authors gratefully acknowledge. We would also like to thank to Prof. Raúl Martins and to Ana Priscila Alves for their support during the development of this work.

References

[1] Fang, Q., Sufi, F., Cosic, I.. A Mobile Device Based ECG Analysis System. InTech; 2014. ISBN 978-953-7619-30-5.
[2] Clarys, J.P., Cabri, J.. Electromyography and the study of sports movements: A review. J Sports Sci 1993;11:379–448.
[3] Stanford, V.. Biosignals Offer Potential for Direct Interfaces and Health Monitoring. IEEE Pervasive Computing 2004;3:99–103.
[4] Lake, S., Bailey, M., Grant, A.. MYO - armband EMG sensor. https://www.thalmic.com/en/myo/; 12-03-2014 (last accessed).
[5] Schwartz, M., Andrasik, F.. Biofeedback: A Practitioner’s Guide. 3rd ed.; The Guilford Press; 2005.
[6] Lourenço, A., Silva, H., Fred, A.. Unveiling the biometric potential of finger-based ECG signals. Computational Intelligence and Neuroscience 2011;2011:1–8.
[7] Agrafioti, F.. ECG in Biometric Recognition: Time Dependency and Application Challenges. PhD Thesis 2011.;
[8] Banz, M.. Getting Started with Arduino. Make Books; 2009.
[9] Bender, P., Kussmann, K.. Arduino Based Projects in the Computer Science Capstone Course. J Comput Sci Coll 2012;27:152–157.
[10] Guerreiro, J., Martins, R., Silva, H., Lourenço, A., Fred, A.. BITalino: A Multimodal Platform for Physiological Computing. Proc of the Int’l Conference on Informatics in Control, Automation and Robotics (ICINCO) 2013;:500–506.
[11] Silva, H., Guerreiro, J., Lourenço, A., Fred, A., Martins, R.. BITalino: A Novel Hardware Framework for Physiological Computing. Proc of the Int’l Conference on Physiological Computing System (PhysCS) 2014;:246–253.
[12] Arduino. Arduino Programming Language Reference. http://arduino.cc/en/Reference/HomePage; 15-11-2013 (last accessed).
[13] Baker, B.. A Glossary of Analog-to-Digital Specifications and Performance Characteristics. Texas Instruments - Application Report 2011.;
[14] Field, A.. Discovering Statistics using IBM SPSS Statistics. 4th edition ed.; Sage; 2013.;
[15] Silva, H., Carreiras, C., Lourenço, A., Fred, A.. Off-the-Person Electrocardiography. In: Int’l Congress on Cardiovascular Technologies. 2013, p. 99–106.
[16] Geddes, L.. The Pruth Shall Set You Free. Development of the Polygraph. IEEE Eng in Med and Biol Mag 2002;21:97–100.
[17] Murphy, J., Gitman, Y.. Open source Pulse sensor. https://www.pulsesensor.com; 12-11-2013 (last accessed).
[18] Silva, H., Alves, A.P., Lourenço, A., Fred, A., Montalvão, I., Alegre, L.. Towards the Detection of Deception in Interactive Multimedia Environments; vol. 7947. HCI and Knowledge Discovery in Complex, Unstructured, Big Data Lecture Notes in Computer Science, Springer; 2013, p. 65–76.