Arduino and Raspberry Pi Based Solution for Electrical Machines Data Acquisition System

Filip Filipović, PhD student, Milutin Petronijević, Assistant professor, Nebojša Mitrović, Full professor and Bojan Banković, Teaching fellow

***Abstract*—Simple data logging and measurements for basic academic lab experiments on electrical machines is usually performed using costly measurement equipment already available in laboratory. On the other hand, non-neglectable impact of low cost open source hardware, like Arduino or Raspberry Pi, has created an alternative for systems with low requirements. Increase in computational power of these devices and rapid decrease in sensors price now result in standalone low power electrical motor acquisition system built for a price just over 100 €. This paper presents one of the solutions of that acquisition system that is intended for student's experimentation with the hardware and software. It adapts to existing sensors of interest in the laboratory to provide the possibility of integration with various sensor and motor types and sizes. The software on the Raspberry Pi 3 is written using Python programming language. Arduino Due is programmed using Arduino variant of C programming language. The overall system design and some sample measurement results, along with GUI (graphical user interface) are shown to demonstrate the usefulness of the system.**

***Index terms*—Raspberry Pi, Arduino, Python, low cost data acquisition system, induction motor**

I. Introduction

Nowadays low cost open source hardware is a very popular topic. Modern prototyping boards are embraced by many universities to tap into students' creativity, as show in [1], provide a low cost experimentation kit, as shown in [2], or develop a real life low cost control system, as shown in [3] and [4]. Putting together different pieces of code and hardware, with the help of a vast community built around these boards, is an idea behind the aforementioned papers. Modern credit size computers (like Raspberry Pi 3 model B) can, in terms of computing requirements for data manipulation and visualization, meet the basic experimentation requirements of electrical power engineering students. Microcontroller boards (like Arduino Due) can do so in terms of data acquisition. Raspberry Pi and Arduino can be easily integrated in MATLAB and Simulink as shown in [1] and [2], they can use some other framework such as [3], be part of larger system using the internet such as [5], or be accessed directly over the internet as shown in [4].

Students often choose Python instead of some other programming languages (such as C++) due to its simplicity, readability and fewer coding hours spent for the same task. The execution speed of a program written in Python is of less importance in this case since the dominant part of it is used for the user interaction. Specialized libraries used for data processing such as Numpy [6] and Scipy [7] are written in lower level programming languages which implies that data will be processed on a speed closer to these programming languages than to Python itself. It is typical for Arduino board based microcontrollers to be programmed using Wirling framework similar to C++. There is an open possibility to write time critical or specific functions in a way that microcontrollers are intended to. The differences between some of the most popular prototype platforms are presented in [8].

The purpose of this paper is to present students with a low-cost data acquisition system which is intended to be used with testing equipment available in the laboratory of electric drives. The primary goal is to make the acquisition system as affordable as possible and the program available in a wide range of data processing devices. They range from desktops, laptops to any other device that runs Windows or Linux operating system. When such a system is established, it can be used for a variety of basic electrical machine diagnostics since it can collect signals from a variety of sensors. It is easily expandable and adjustable to currently not supported dominant sensor types due to circuit for sensor signal scaling and shifting consisting of operational amplifiers and other passive electrical components.

This paper is structured as follows: section 2 gives a detailed description of used hardware components along with justification of each choice and cost specification; section 3 presents demonstration results and section 4 discusses conclusions and future work.

II. Materials And Methods

The base idea was that the system by itself has to be applicable for conducting data acquisition and data analysis without any third-party computers or software. Software libraries that will be used have to be free of charge for both personal and commercial application.

Filip Filipović is with the Faculty of Electronic Engineering, University of Niš, 14 Aleksandra Medvedeva, 18000 Niš, Serbia (e-mail: filip.filipovic@elfak.ni.ac.rs).

Milutin Petronijević is with the Faculty of Electronic Engineering, University of Niš, 14 Aleksandra Medvedeva, 18000 Niš, Serbia (e-mail: milutin.petronijevic@elfak.ni.ac.rs).

Nebojša Mitrović is with the Faculty of Electronic Engineering, University of Niš, 14 Aleksandra Medvedeva, 18000 Niš, Serbia (e-mail: nebojsa.mitrovic@elfak.ni.ac.rs).

Bojan Banković is with the Faculty of Electronic Engineering, University of Niš, 14 Aleksandra Medvedeva, 18000 Niš, Serbia (e-mail: bojan.bankovic@elfak.ni.ac.rs).

Unlike [9] where Arduino Pro Mini is used, larger memory and ADC (analog to digital converter) with resolution higher than 10 bits are desired for basic electrical machines measurements. For that reason, cheaper Arduino Pro Mini was replaced with more powerful Arduino Due which is based on 32-bit SAM3X8E ARM Cortex-M3 CPU. Arduino Due uses 4 of his 12-bit ADC for sensor inputs and two of its digital inputs (pins 2 and 13) for hardware quadrature encoder input. Communication with it is done via USB (universal serial bus) port and in this configuration Raspberry Pi 3 model B is the device that sends the commands and receives the results. The program is written so that any Windows or Linux based device can run it and control Arduino. The overview of the system is shown in Fig. 1. The two platforms supplement each other. Raspberry provides a base for data storage, visualization and user interaction, and its lack of analog inputs is compensated by Arduino.

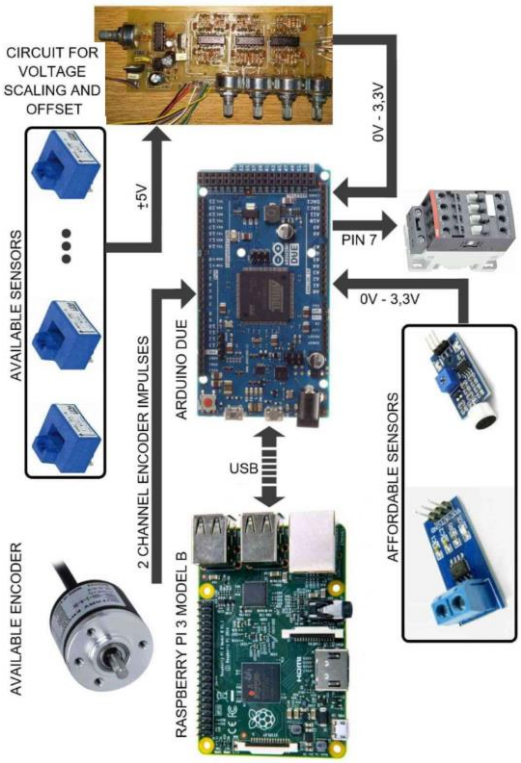

Fig. 1. Overview of the low-cost data acquisition system

Voltages up to 3,3 V are allowed on the input of the microcontroller and higher than that can damage the microcontroller board. To adapt to the existing sensors in the laboratory, which provide output of up to ± 5 V, circuit for signal scaling and offset had to be built. In addition, since some sensors wires can have substantial length and inappropriate shielding, a non-neglectable amount of noise can be picked up along the way, especially if the wires pass by experimental motor drives. To suppress high frequency noise, a low pass first order filter is designed with cut off frequency of 10 kHz (simple RC circuit). Along with filtering, there is a possibility of offsetting output signal up to 2,5 V, and that is achieved for all 4 channels simultaneously with one potentiometer. Signal gain for each channel can be defined independently with 4 potentiometers and the maximum gain is 1 (that configuration without offset can be used only if signal filtering is required). The circuit is supplied with 12 V DC. To achieve potential negative output voltage, the known configuration for voltage inversion using 555 timer integrated circuit is used for operational amplifier negative supply pin (see Fig. 2 for detailed look).

Up to 40000 points can be recorded by the Arduino and then transmitted back. The user choses channels of interest, the sampling time of each channel (from as low as 10 μs) and the total number of points to be recorded. This implies that problems of overcoming relatively slow sampling times for more precise condition monitoring considered in [10] can be reduced to a suitable level for demonstration purposes. After these choices, the program will automatically arrange the number of points for each channel so that the data acquisition has the same time span. If the encoder input is used, the user must define the number of impulses per rotation of the encoder. There is an option to activate one digital output (digital pin 7) when the acquisition starts. Some transient states (such as induction motor direct on line start) can be recorded more easily if the signal of that pin is used for motor contactor close command. After that, the user is asked to enter the name of the experiment that is conducted and it is packed in .csv file so that it can be easily analyzed later. After that, the number of acquisitions and the pause between the two consecutive ones is defined. Moreover, there is an option for continuous acquisition mode, where next acquisition cycle will start immediately after the recorded data is transmitted to Raspberry. When the folder where the results are saved is selected, the user can start acquisition. Each channel in each cycle will have a separate .csv file and aforementioned .csv will keep track of the recorded files and their synchronization for further user data manipulation. The first column in .csv data record contains time, and the second contains the recorded parameter. Files can be analyzed in some other software (such as MATLAB or Mathematica) and the headers that are present in each file, and define the name of each column, can be useful when doing this task automatically.

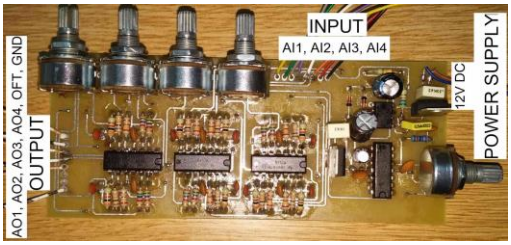

Fig. 2. Detailed look on circuit for voltage scale and offset

Since the recorded files are just arrays of numbers, gaining a good and quick grasp can be challenging for most people. People often prefer some sort of visualization over plain number representation. One part of written software has a job of visualizing recorded experiments of interest, and library used for visualization is Matplotlib [11]. It is only the measured speed transmitted back from Arduino Due which has the value that represents the actual speed. All the other recorded data is, for the sake of faster data transfer,

transmitted as raw analog input read (values from 0 to 4095). Data manipulation (scaling and filtering using Kalman filters) is done using Numpy and Scipy libraries. After the appropriate manipulation, the data can be saved as a separate file so that it can be accessed directly the next time it is needed.

TABLE I
LIST OF MATERIALS

Component	Price [€]
Raspberry Pi 3 Model B	47
Arduino Due	26
Circuit for voltage scale and offset	18
SD card (16 Gb)	8
Power supply (5 V)	5
Arduino Hall based current sensor (30 A)	4
Arduino sound sensor	5
Total:	**113**

Addressing the issue of low cost induction motor acquisition system cannot be completed if its price is just too high, the components used are hardly commercially available or if complicated programming and wiring are required. Table I gives a list of materials with the cost of the used components. What is used for the price calculation are average prices on internet shops available in Serbia (March 2017).

III. MAIN RESULTS

Parts of the finished program will be shown and the focus will be on data visualization, as of something that will be used as such at the beginning of experimentation with electrical machines. It is important to point out not only that the software can work on most Windows or Linux systems, but also the possibility to communicate with any serial device (not only Arduino Due). It also acquires any sort of data as long as there is a strictly defined way of exchanging it. However, for now there is only a defined way of exchanging data with Arduino Due. The data can be visualized in the form of trace graph, Fourier transformation or spectrogram.

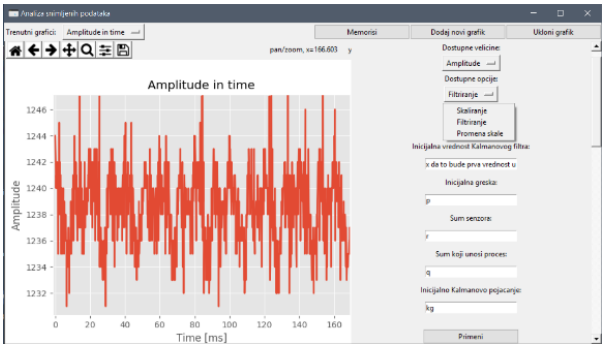

Fig. 3. Zoomed trace graph of recorded data

In many applications a trace graph does not provide enough useful information by itself. Such an example could be the case of motor vibration analysis, where just the amplitude of vibration does not necessarily provide useful information. A much more desirable combination is that of present frequencies and their amplitudes.

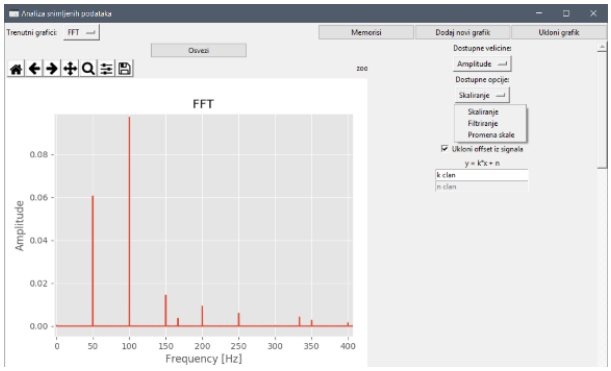

Fig. 4. Fourier transform of recorded data

Although Fourier transform can be useful for inspecting present frequencies in data, it cannot provide the information which frequencies were present at each moment. This is a very useful feature for motor vibration analysis because there can be a set of frequencies with fade in and fade out effects. If the detection of those frequencies is attempted with standard Fourier transform, they can be much harder to see. For that reason, another useful tool, spectrogram, is used.

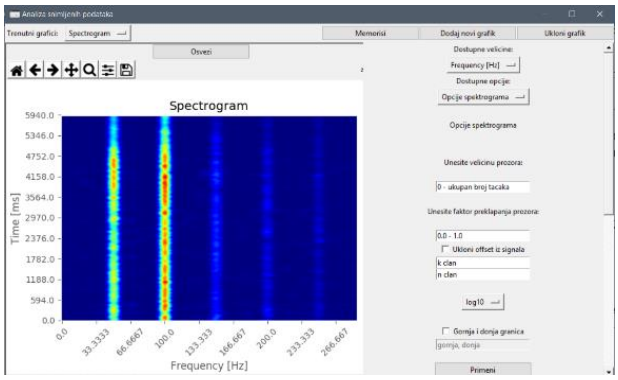

Fig. 5. Spectrogram view of recorded data

The bottleneck of the system is serial communication. If a faster sampling rate and live representation of recorded data are desired, serial communication becomes a limiting factor. Serial communication, in general, does not include only recorded data, but also data labelling and blocks for data integrity check, like CRC (cyclic redundancy check), as shown in [9]. Furthermore, all that increases the time of data transfer and the lower limit of data sampling time that can be seen in real time. The fact that there can be up to 4 channels that record data, all with different sampling time, further expands the structure of the transmitted message. The data, in general, must be stored in Arduino and then transmitted to Raspberry. This implies that signal sampling can be done for

only a finite period of time, after which the recorded data has to be transferred. Even if a new cycle starts immediately after the transfer, there is a non-recorded window of time when data was transferred.

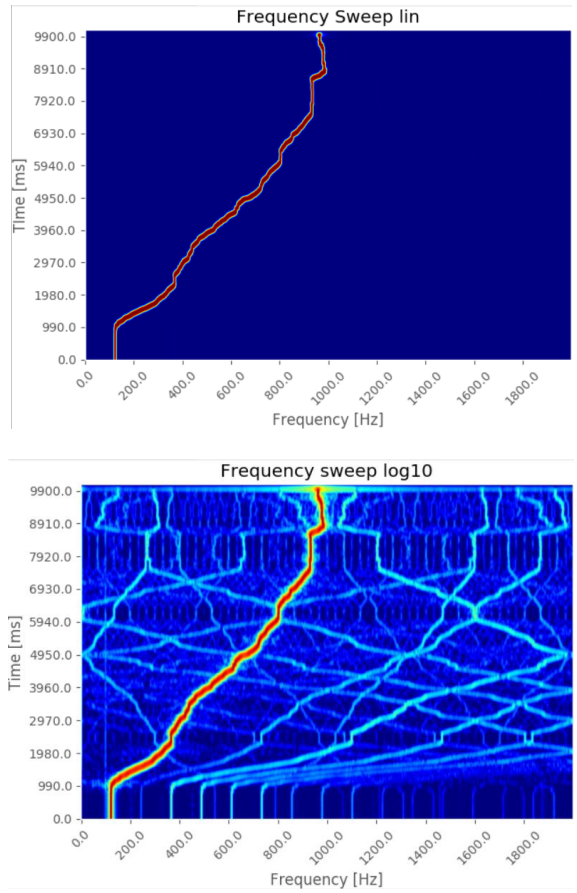

Fig. 6. Spectrogram visualization of frequency sweep, using linear (upper figure), or logarithm (lower figure) amplitude scale.

"It costs a little, but how worth is it?" is an obvious question and the answer to it will come in a form of comparison. Function generator "ISKRA MA3732" is used to provide various signal waveforms, amplitudes and frequencies. The signal is fed into Arduino and into oscilloscope "Tektronix DPO3034" for comparison. After those measurements, both devices were recording line current of induction motor driven by variable frequency drive, oscilloscope using his own current clamps and Arduino using cheap Hall effect based current sensors. When the signal generator output voltage was recorded, it was fed into Arduino over aforementioned circuit for scaling. In this configuration there was no galvanic isolation for measuring voltage, although there is possibility for it if appropriate sensors are used.

In the first test, frequency sweep was done with a sinusoidal signal provided by the function generator, and it was not very uniform due to the fact that it was done by hand. The potentiometer was being turned from the lower limit (100 Hz) to upper (1000 Hz) over a period of 10 seconds. As it can be seen in Fig. 6. linear representation of data gives distinguishable dominant frequency, while lower magnitude frequencies can be hard to detect. Logarithmic representation of the recorded data can give an insight in other present frequencies of much lower magnitude and it is up to the user to define a preferred way.

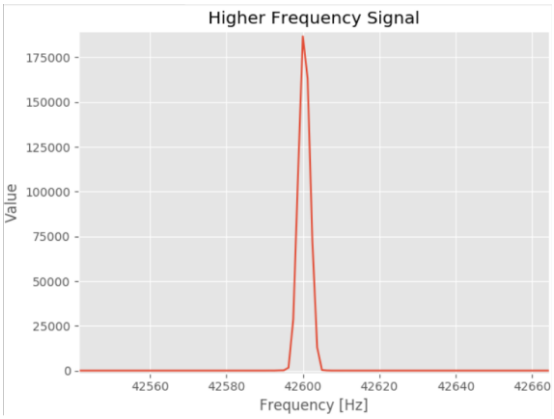

Fig. 7. Fourier transform of recorded signal containing high frequency signal for Arduino Due upper frequency limit test.

The next test was upper frequency limit of the device. As mentioned before, the minimum sampling time of Arduino Due was set to be 10 µs. It implies that the upper limit of the frequencies to be seen with it is 50 kHz. For this test, the frequency of just over 42 kHz was used and the magnified part of Fourier transform graph was shown in the Fig. 7. There were no other noticeable frequencies and the transformation was done on raw data (not scaled to match the input voltage).

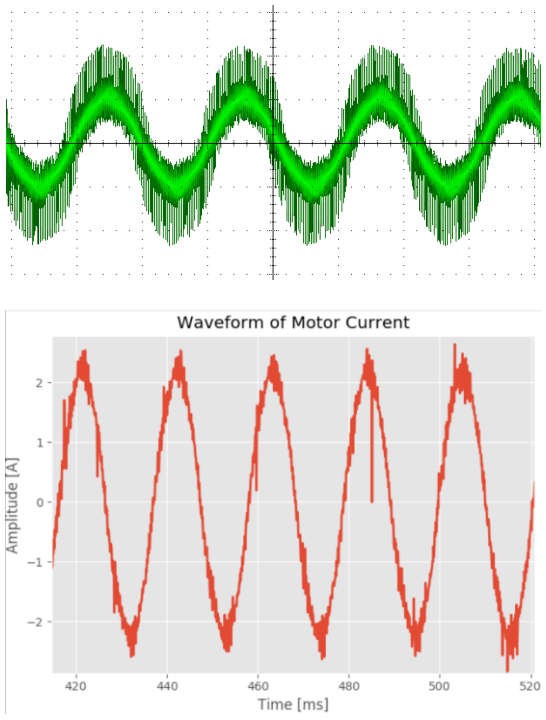

Fig. 8. Motor current waveform recorded using oscilloscope (upper figure) and Arduino Due (lower figure)

A combination of electric motor and variable frequency drive was used for the last set of tests. The purpose is not only to observe the results and see the usefulness of the data, but also to observe the system itself. In other words, the influence of all disturbances on it. If the system is not up to task, further hardware or software changes have to be made until the primary goal is achieved. Comparison of recorded current using oscilloscope and Arduino can be seen in Fig. 8.

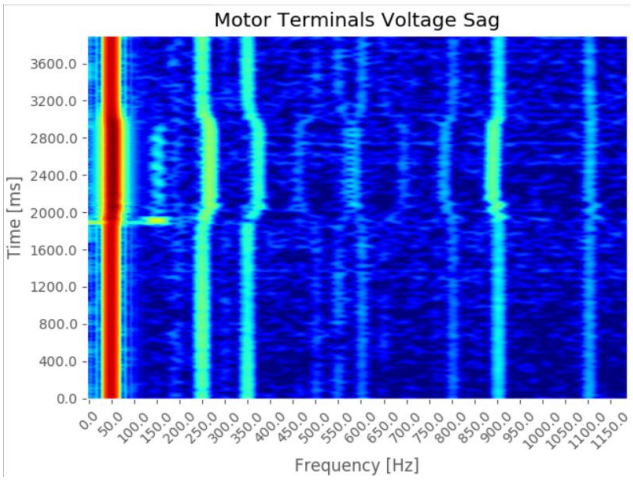

Fig. 9. Motor terminals voltage sag (at around 1,9 s) that lasted 1 s and its influence on harmonics present in line current

When Arduino Due is in the immediate proximity of variable frequency drive or induction motor power supply wires, there is noticeably more noise present in the recorded data (30 units peak-to-peak compared to 7 units in a low noise environment on 4096 units full resolution that correspond to 3,3 V). As the last experiment, the influence of induction motor terminal voltage sag on motor current harmonics was recorded. The sag occurred about 1,9 seconds after the data recording had started and it lasted for about a second. Discussions on occurred harmonics can be a topic for some different papers and this one only shows current spectrogram for illustrative purpose (Fig. 9.). The warmer colours in the spectrogram are used for points of highest amplitude. Points of lowest amplitude are coloured with dark blue colour. Exact value of each point can be obtained by hovering over the point of interest with mouse pointer.

IV. Conclusion

All of these would be modest tools for electrical machines scientist, but good enough for an undergraduate electrical power engineering student. Recording present frequencies in magnetizing current of transformer or similar level experiments in laboratory can be done by this system. The simplicity of the program can be an advantage for someone who just wants to see some signal without doing daunting system setup.

Free software libraries usually come with a price. Because they usually rely on some community of volunteers to update and fix bugs of those libraries, no one guaranties that the provided code is completely tested. The software of similar functions could have been made using MATLAB or LabVIEW development environments. Time of development and testing would have been arguably much shorter, and provided functions probably better tested and optimized. Mathworks Simulink and LabVIEW provide a possibility of graphical programming. This is a very convenient feature for an engineer with no programming background or for a rapid system prototyping. These environments offer hardware support for popular boards, and packages for optimized code generation. This implies that easy setup and generated code of decent performance will be provided without complicated setup and tuning. Only drawback can be the fact that these environments are not free, and so, modification of the software by a student could be possible only if he owns appropriate licence.

As it was pointed before, these boards have novice friendly vast community built around them. On the other hand, boards such as BeagleBone Black where usually the community assumes some advanced level of Linux programming knowledge was the reason previous combination was used. However, BeagleBone Black is arguably a better solution for this purpose when it comes to experienced microcontroller programmers. Designing a top-notch, but overwhelmingly complex system from a programmer's standpoint, was not the idea in the first place. Potentially conducting a substantial number of successive acquisitions and storing them in an ordered and labelled way is the program feature that will be used for future machine learning applications in areas of classification and anomaly detection.

References

[1] Jamieson, Peter, and Jeff Herdtner. "More missing the Boat—Arduino, Raspberry Pi, and small prototyping boards and engineering education needs them." *Frontiers in Education Conference (FIE), 2015. 32614 2015. IEEE*. IEEE, 2015.

[2] Reck, Rebecca M., and Ramavarapu S. Sreenivas. "Developing a new affordable DC motor laboratory kit for an existing undergraduate controls course." *American Control Conference (ACC), 2015*. IEEE, 2015.

[3] Sobota, Jaroslav, et al. "Raspberry Pi and Arduino boards in control education." *IFAC Proceedings Volumes* 46.17 (2013): 7-12.

[4] Reguera, P., et al. "A low-cost open source hardware in control education. case study: Arduino-Feedback MS-150." *IFAC-PapersOnLine* 48.29 (2015): 117-122.

[5] Ferdoush, Sheikh, and Xinrong Li. "Wireless sensor network system design using Raspberry Pi and Arduino for environmental monitoring applications." *Procedia Computer Science* 34 (2014): 103-110.

[6] "NumPy — NumPy", *Numpy.org*, 2017. [Online]. Available: http://www.numpy.org/. [Accessed: 01- Apr- 2017].

[7] "SciPy.org — SciPy.org", *Scipy.org*, 2017. [Online]. Available: https://www.scipy.org/. [Accessed: 01- Apr- 2017].

[8] Maksimović, Mirjana, et al. "Raspberry Pi as Internet of things hardware: performances and constraints." *design issues* 3 (2014): 8.

[9] Alves, Ana Priscila, et al. "BITtalino: A Biosignal Acquisition System based on the Arduino." *BIODEVICES*. 2013.

[10] Sapena-Bano, A., et al. "Condition monitoring of electrical machines using low computing power devices." *Electrical Machines (ICEM), 2014 International Conference on*. IEEE, 2014.

[11] "Matplotlib: Python plotting — Matplotlib 2.0.0 documentation", *Matplotlib.org*, 2017. [Online]. Available: https://matplotlib.org/. [Accessed: 01- Apr- 2017].