Preprints of the 10th IFAC Symposium Advances in Control Education
The International Federation of Automatic Control
August 28-30, 2013. University of Sheffield, Sheffield, UK

Control Practices using Simulink with Arduino as Low Cost Hardware

R. Barber, M. Horra. J. Crespo.

University Carlos III of Madrid. SPAIN

(e-mail: rbarber@ing.uc3m.es, mario.horra@alumnos.uc3m.es, jocrespo@ing.uc3m.es)

Abstract: This work describes a Simulink lab practice using Arduino as low cost hardware. A Shell must be developed in order to adapt Arduino signals to the real plant, consist of a DC motor. With Arduino architecture and with open hardware a cheap Data Acquisition card has been build. Several tests have been done to validate de full system and a frequency study has been completed in order to know the possibilities of the proposed architecture in the control of new other plants.

Finally, a lab exercise using the Simulink and Arduino system is proposed to the student of engineering bachelors. The exercise consisting of apply the empirical method of Ziegler-Nichols for adjust a PID controller, testing this controller over the experimental plant.

Keywords: Simulink, Arduino, DAQ, Control labs.

1 INTRODUCTION

The main goal of this work is to use low cost hardware, to connect Simulink with a real system for lab practices in Control Engineering bachelors at Carlos III University of Madrid. Nowadays, the practical prototypes are connected to a computer through expensive DAQs. When computers or DAQs are updated, it implies a very high cost. In the case of internal DAQs, connected on an internal bus of the computer, the difficulties are bigger. New Arduino based DAQ must be adapted to the lab prototypes in order to avoid hardware changes or modifications on it.

1.1 Simulink and low cost hardware

Arduino is an open hardware platform utilized along this project. All necessary tools or documentation to work with this system can be found on its webpage (Ditecom Design S.L, 2009). There is also an active forum (Keithley Instruments, Inc, 2013) where the community around this technology shares knowledge. Additional libraries and documentation can be located in (Kyhe Ingeniería, S.L., 2013) and (National Instruments Corporation.,2013) All of this make that Arduino is constantly growing up and establishing as a valuable technology in educational institutions as well as in little and small scale engineering solutions.

Matlab and Simulink have been consolidated in engineering and in academic world as a design and development tool. So the ability of linking both platforms is an interesting matter.

There are two ways to communicate Arduino with Matlab Simulink: the new one consists of installing the support target for Simulink that comes with Matlab 2012b version, Fig. 1, and requires Windows operating system; if that version or Windows is not available in the computer, the other way is about downloading the Arduino IO package from the file exchange area of the mathworks web. It is a good option to run it on Linux or Mac computers.

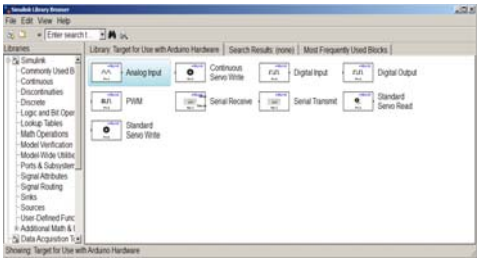

Fig. 1. Arduino library for Simulink.

The manner they work is different in each case. Arduino IO package is based on a server running on the Arduino microcontroller that listens for Matlab commands from the serial port, and after execute such commands, if asked, returns an answer.

However, the other method consists of converting the Simulink model to code that runs directly on Arduino, therefore, the microcontroller can be disconnected from host computer. Theoretically, this second manner should allow a faster sample rate, but as we will see in the sampling rate analysis section, working in external mode make this form slower.

The idea of this work is to hide to the student the way the computer works with the real plant, being possible work with Simulink libraries in a Simulink model, or working as a DAQ in which the microcontroller only is a link between Simulink and the plant.

1.2 Data acquisition systems

Data acquisition (DAQ) deal with taking samples from real world (Fig. 2) to generate data that can be processed for a computer program or by any other device that works with digital signals (Arduino, 2013c). Measures obtained through sensors are analogic as the real world nature, and these electrical signals must be converted into digital ones. Applications in different fields where is necessary to work with real world measures are numberless. Representative cases are the employment of DAQs to build optical telescopes to detect cosmic rays (Arduino, 2013b), or in CERN experiments as LHCb (Arduino, 2013c) or COMPASS (B. W. Evans, 2012). As is represented in the flux diagram of information (Fig. 2) Analog sensors feel the world, and its analogic magnitudes go through a data acquisition step to generate the information that can be understood by the most of electronic devices. Nowadays, a lot of DAQ systems can be found in the market (R. Pallás, 2005) (J. Boyer, 2002) (G. Haefeli, 2006) (L. Schmitt. 2004), although cost is significantly higher than the hardware and software implementation described in this work.

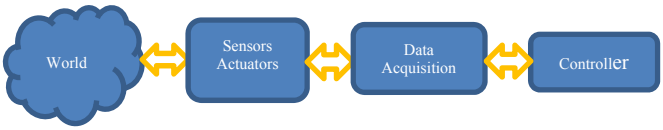

Fig. 2. Connection between world and controller.

2 DATA ACQUISITION SYSTEM DESIGN

The goal of this work is to control a DC Motor with Simulink applied to practice labs for control engineering bachelors. To connect Arduino to the practice model, signals must be conditioned. In order to get it, an Arduino shield has been designed.

2.1 Data Acquisition System Requirements

The hardware interface between Arduino and the practice plant must take into account the characteristics of the input and output levels of the microcontroller and of the practice model.

The input of the prototype, the reference signal, consists of a continuous voltage signal in the 0-10V range. However, at the output of the Arduino hardware, we have a PWM signal in the 0-5V range. Although a DC motor can be controlled directly with a PWM signal a filter ha ben included in order the use of other analog systems that not works with PWM.

To close the control loop, we are sampling the velocity of the motor. The limits of this signal are [-5, +5] V, that are also not suitable to the Analog Input of the Arduino that is constrained to [0, 5] V.

2.2 Model input signal conditioning

Once that Arduino generates the PWM signal constrained to [0, 5] V, it is necessary to adapt it to the [0, 10] V required by the experimental model. To adapt the signal the adequate electronics has been designed, containing a filter to obtain a continuous one and an amplifier to adapt voltage range, as is shown in Fig. 3.

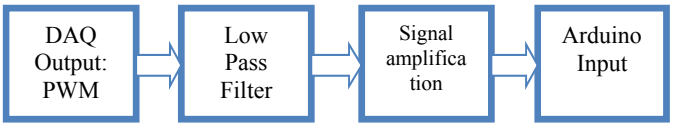

Fig. 3. Connection between world and controller.

PWM conversion signal to analog one requires a compromise between cost, low cut-off frequency, and low ripple on the DC output signal. To reach such specifications, a low pass, passive and simple pole filter is implemented. To minimize ripple and reach a low response time a 1μF capacitor and a 15kΩ resistance are selected. These values provide a 10.61Hz cut-off frequency, and a ripple voltage of $0.17V_{pp}$. Filter transfer function is indicated in equation (1),

$$G(s) = \frac{\frac{1}{s \cdot RC}}{1 + \frac{1}{s \cdot RC}} = \frac{66.67}{1 + 66.67} \quad (1)$$

In Fig. 4 frequency response is represented. The time response obtained is 0.034s, quite enough to deal with model laboratory DC motor.

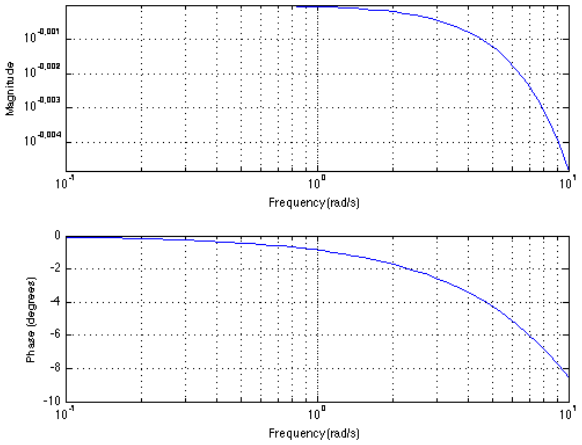

Fig. 4. Frequency representation of the filter

Once signal is filtered, its range must be adapted to the input range of the motor. For that purpose, an operational amplifier in non inverting configuration will be used at the output of the filter. In Fig. 5 signal conditioning from the PWM signal of the microcontroller to the reference signal of the motor can be observed.

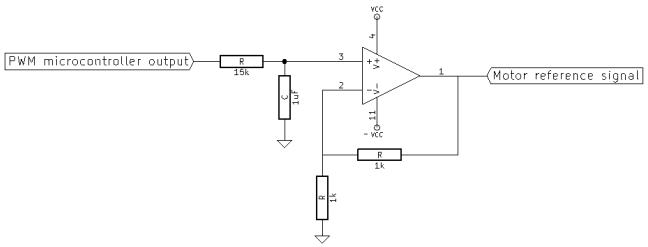

Fig. 5. PWM signal conditioning electronic scheme.

Actually, feedback resistance could be set to a variable one to adjust Arduino and plant interface.

2.3 Model output signal conditioning

Model has an encoder to measure the position of the motor axis that outputs a voltage signal in a range between [-5, 5]V. This signal must also be conditioned to the microcontroller input between [0, 5]V (Fig. 6). To do that, a voltage divider is placed at the output of the model output signal that set that signal in the [-2.5, 2.5] V range. After that an offset is applied to get the desired [0, 5] V.

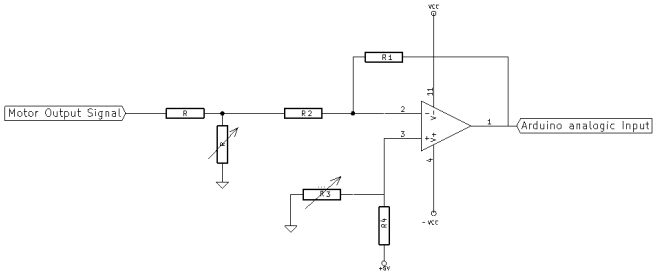

Fig. 6. Model output signal conditioning electronic scheme.

3 PRELIMINARY TEST

In order to validate the proposed DAQ system, some preliminary tests have been done using Simulink with Arduino library. Several experiments have been done: study of step and sine signals to validate the developed PWM filter and study of the full system consist of closing the loop using the practice model.

3.1 Step signal

In the first test Arduino has been connected to a pulse generator and a constant as an offset, as is shown in Fig. 7.

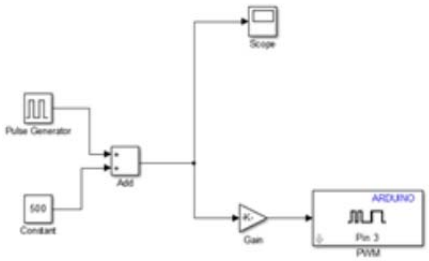

Fig. 7. Simulink model for step test.

The goal of this first experiment is both to test the Simulink connectivity with the Arduino and the quality of the PWM to analogic filter build. The quality of the signal is fairly good to work with the proposed platform. Fig. 8 shows the PWM signal and the analogic one after the filter in this experiment.

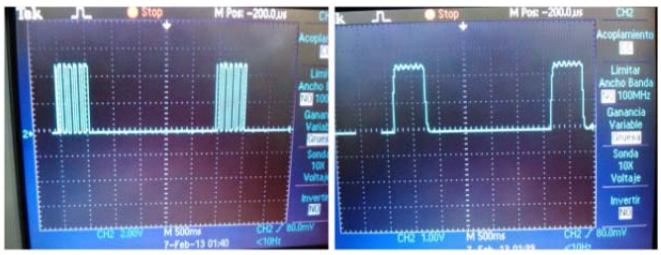

Fig. 8. Step signal before and after the filter.

Fig. 9 shows a plot of the obtained step signal, in which noise can be appreciated.

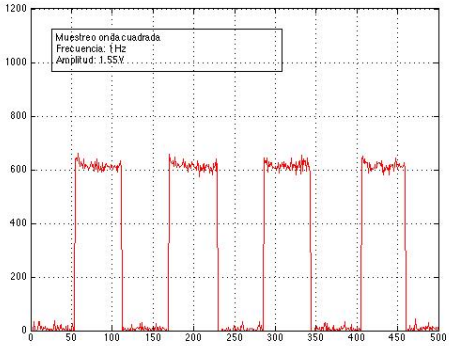

Fig. 9. Signal detail after filtering.

3.2 Sine signal

In the same way, Arduino has been connected to a sine generator and a constant as an offset, as is shown in Fig. 10.

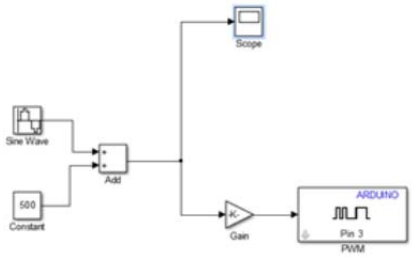

Fig. 10. Simulink model for the sine test.

The goal of this first experiment is also to test both, the Simulink connectivity with the Arduino and the quality of the PWM to analogic filter build again. In this case, the quality of the signal must be better after the PWM due signal characteristics. Fig. 11 shows the PWM signal and the analogic signal after the filter in this experiment.

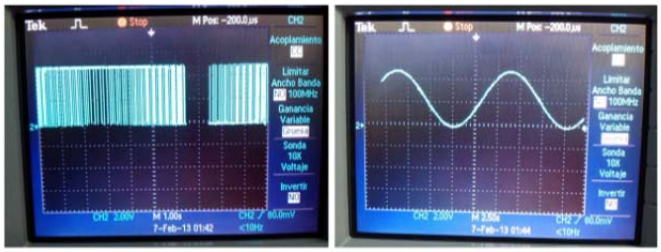

Fig. 11. Signal before and after the filter.

Fig. 12 shows the plot of the obtained sine signal, in which noise can be seen. In this case, it can be appreciated that the filter works better, due the smoothness of the signal, and noise value is reduced comparing to the previous experiment with the step signal

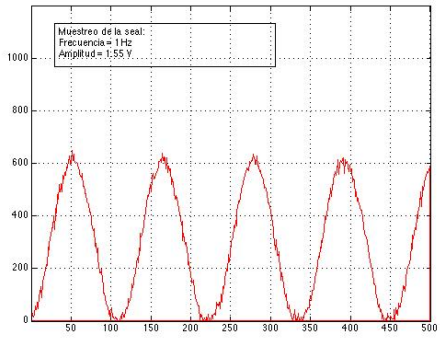

Fig. 12. Sine signal detail after filtering.

3.3 System response

At last, a complete test will be carried out over the whole system constituted by the Simulink controller, Arduino board, signal conditioning and the real plant. In the next figure, Simulink model is shown,

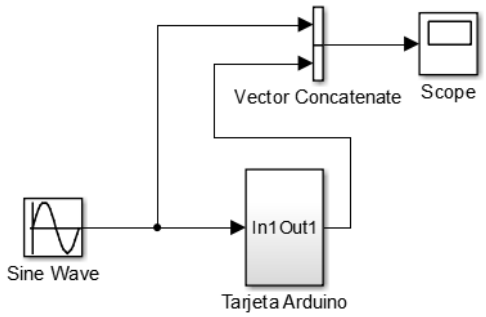

Fig. 13. Simulink model to test the system response.

The clock model named Arduino target contains the Simulink blocks from the Arduino library that connect to the plant. In Fig 14,

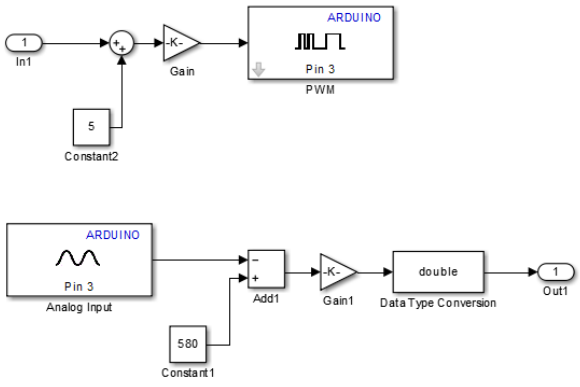

Fig. 14. Simulink block with the Arduino library blocks.

Finally, in Fig: 15, the result of the experiment is shown. To this test a sine signal and the speed of the motor as output signal have been chosen. In the figure the output signal in red colout is amplified and delayed, as corresponding with a first order system.

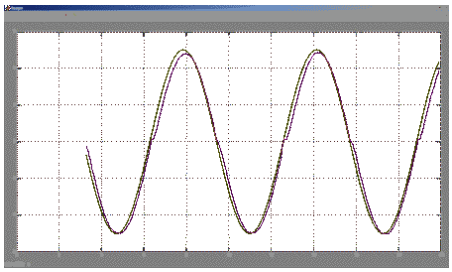

Fig. 15. Signal response using a sine signal as input.

4 SAMPLING RATE ANALYSIS

In order to know the possibilities of the proposed architecture to in the analysis and control other mechanical, electrical or robotics plants, a frequency study has been done. With this study we can conclude the range and frequency range in which we can sample and work with a real system.

To do the study, it is necessary differentiate the two methods of working with Simulink and Arduino.

4.1 Arduino support from Simulink

As it has been commented, Arduino support from Simulink that makes the algorithm runs in the microcontroller contains a library of Simulink blocks, where there is one called “Analog Input”, Fig 16, which is in charge of measuring the voltage of an analogic pin. This Simulink block allows setting the sample rate to theoretically a value as low as 0.000001s, but such a value is sure to frozen most of computers. A reasonable value is 0.02s sample time; lower values mean a change in scale time axis.

Fig. 16. Simulink model test for the frequency test.

According to this schema, several experiments with different sample time were done. Tests consist of a square wave of 1Hz frequency in one Analog pin of the Arduino. In Fig. 17 results using a sample time of 0.02 is shown.

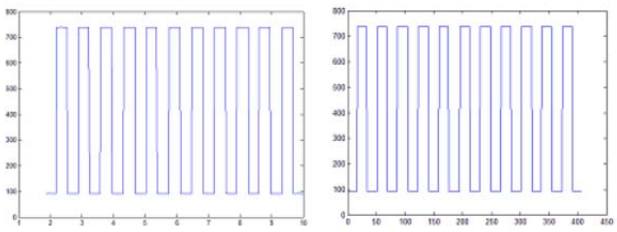

Fig. 17. Signal vs. time (seconds) vs. sample number. Sample time = 0.02s.

As can be observed in the figure, in this experiment, the microcontroller has taken exactly 407 samples, which actually means that sample time is 0.024s.

To amplify this effect, we have repeat the test using as sample time t=0.002s. In Fig. 18 results using this new sample is shown.

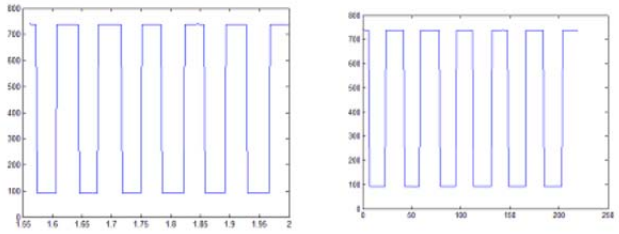

Fig. 18. Signal vs. time (seconds) vs. sample number. Sample time = 0.002s.

In this case, it has registered 220 samples in 5 seconds, that give us a sample time = 0.022s. Looking in the time scale of the Figure 8, and according to our 1Hz signal, obviously this means there is not real time here. And we have a sample time that depends on the program loaded and can vary ±10% around 0.02s. We can deal with it for students' practices as we get this system at such a low cost.

4.2 Arduino IO package

In this case, the microcontroller only contains a server that is running continuously. The Simulink schematic will also be built with an Analogic Read and a scope, as is shown in Fig. 19.

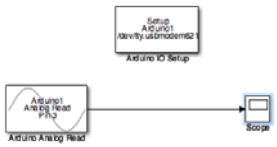

Fig. 19. Simulink blocks for Arduino IO package.

As in previous case, 1Hz frequency square wave connected to an analogic pin of the Arduino is used.
Now, in this case 1001, Fig. 20, samples are registered in 10s, meaning that it has reach the sample time. Also, there is not real time, and if sample frequency is increased, the same reduction will occur in the time scale. But in this case, as the program running on the microcontroller is always the same, the sample rate is constant at 100 samples per second.

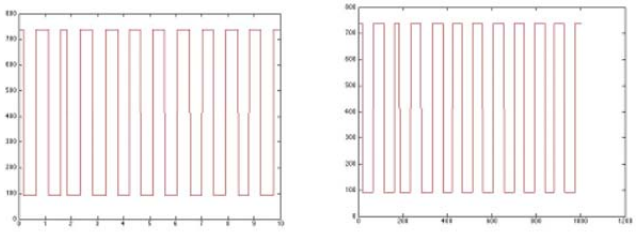

Fig. 20. Signal vs. sample timeand vs. sample number. Sample time = 0.01.

5 LAB PRACTICES

Finally, a full practice for the students is developed and tested in the practical platform. The practice consist of adjust and test the Ziegler Nichols method to adjust a PID controller.

5.1 Practice description

The real plant has a DC servomotor with maximum speed 4000 rpm and 24Vcc. It has a tacho generator and an encoder as speed and position sensors, respectively. Fig. 21 shows experimental platform.

Fig. 21. Experimental servomotor platform.

In a first step, student must close the loop in order to get continuous oscillations to calculate the parameters of the PID controller.

5.2 Practice description

In a first step, student must close the loop with a proportional regulator in order to get continuous oscillations to calculate the parameters of the PID controller.

The student must connect input and output of the Arduino based DAQ developed in this work and makes the Simulink model shown in Fig. 22.

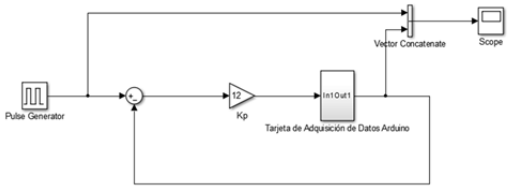

Fig. 22. Simulink model for gain adjustment.

As the student goes increasing the parameter of the proportional regulator and reaches the critical gain, the system is going to show constant amplitude oscillations. (Fig. 23)

Fig. 23. Oscilation output signal from the motor.

For this critical gain (K_c=12), the period of oscillations is measured: T_c= 0.14s

According to the values proposed by Ziegler-Nichols in the closed-loop tuning method, regulator parameters are calculated:

$$K_p = 0.6 \cdot K_C = 7.2$$
$$K_i = 1.2 \cdot \frac{K_c}{t_c} = 11.9$$
$$K_d = 0.075 \cdot K_c \cdot t_c = 0.126$$

To test how the control system works, students should build the Simulink scheme, Fig 24, which includes proportional, derivative and integral parameters of the PID regulator and introduce calculated gains as is presented in next figure, that includes model and Arduino interaction block.

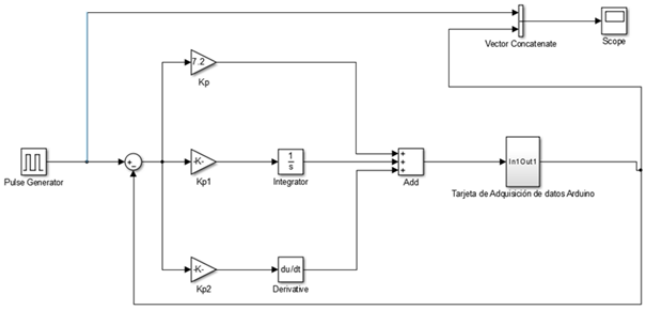

Fig. 24. Simulink model for the PID controller.

Once control scheme have been built, students can validate it. (Fig. 25) In Fig 25 action control over the model can be observed,

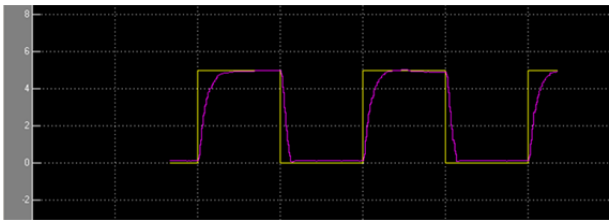

Fig. 25. Output of the controlled plant.

In Fig. 26 a response detail is shown

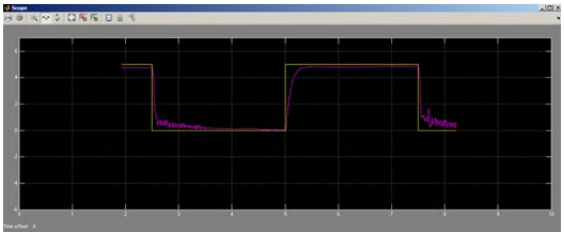

Fig. 26. Output signal detail

Once the regulator is shown, students are encouraged to test different values for the parameters and see the effect of each one. Especially to analyze how depending on the derivative one, they can get a slower and more stable response, or a rapid and less stable one. And the integral parameter influence to reach the reference signal.

6. CONCLUSIONS

In this work a low cost control system based on an Arduino microcontroller have been described. Arduino works as a DAQ between Simulink and a real plant. An Arduino shell has been designed and developed in order to adapt signals from the microcontroller to the experimental plant.

Several tests have been done to validate de full system. A frequency study has been done too in order to know the possibilities of the proposed architecture in other plants. Frequencies reached allow using it in most of temporal studies of mechatronics systems. Finally, a practise exercise has been proposed, solved and tested in order to validate the proposal.

REFERENCES

Arduino (2013a). http://arduino.cc/es/Main/Software Last vie: March, 2013.
Arduino. (2013b). http://arduino.cc
Arduino. (2013c). http://arduino.cc/es/Tutorial/HomePage. Last view: March 2013.
B. W. Evans. (2012) Arduino programming.
R. Pallás. (2005). *Adquisición y Distribución de Señales*. Editorial Marcombo.
J. Boyer, B. C. Knapp et al. 2002. *FADC-based DAQ for HiRes Fly's Eye*.
G. Haefeli, A. Bay, A. Gong, H. Gong, M. Muecke, N. Neufeld, O. Schneider . 2006. *The LHCb DAQ interface board TELL1* **.**
L. Schmitt. 2004. *The DAQ of the COMPASS experiment*.
Ditecom Design S.L. (2009). http://www.ditecom.com/instrumentacion/tarjetas-adquisicion-datos.shtml
Keithley Instruments, Inc.(2013). http://www.keithley.com/products/data/multifunction/usb/?mn=KUSB-3100. Last view: March 2013.
Kyhe Ingeniería, S.L. (2013) http://www.kyheingenieria.com/catalogo29.htm. Last view March 2013.
National Instruments Corporation. (2013) http://www.ni.com/data-acquisition/esa/. Last view March 2013.