DEVELOPING SKILLS IN ENGINEERING CAPSTONE PROJECTS WITH LOW-COST MICROCONTROLLER SOLUTIONS: THE EPS@ISEP EXPERIENCE

P. Ferreira[1], B. Malheiro[1,2], M. Silva[1,2], J. Justo[1], P. Guedes[1], A. Duarte[1,3], C. Ribeiro[1,4]

[1]Polytecnic of Porto, School of Engineering (PORTUGAL)
[2]INESC-TEC Porto (PORTUGAL)
[3]REQUIMTE/LAQV Porto (PORTUGAL)
[4]INEB/I3S Porto (PORTUGAL)

Abstract

The European Project Semester (EPS) project-based learning framework is a multicultural and multidisciplinary one semester engineering capstone programme provided by a network of European Higher Education institutions. Its aim is to prepare 3rd-year undergraduate students to their future professional life, enhancing hard and soft skills and following ethical and sustainable design and development practices. At the School of Engineering of Porto Polytechnic (ISEP) the focus of the EPS programme (EPS@ISEP) is on solving multidisciplinary problems through teamwork, involving engineering, design and business students [1].

The students work in teams of 5 to 6 students, assembled according to the identified Belbin team roles, and also maximizing student cultural and scientific diversity. On the first week each team chooses to solve one of the open-ended multidisciplinary problems on offer. Those projects involve typically some type of automation and control[2].

One of the obstacles these eclectic teams face is the lack of hardware/software skills required to design, assemble and test a microcontroller based systems. To help overcome this situation, the programme syllabus includes an 8-hour intensive "Arduino & Electronics Crash Course" at the beginning of the semester due to its market penetration, low-cost, availability of documentation and support, and soft learning curve. This course has effectively contributed to provide students with the necessary knowledge to design and implement simple control systems, leading to the adoption in multiple EPS@ISEP past projects of the Arduino platform/ecosystem. However, the crescent sophistication of the projects, namely the integration with Internet of Things (IoT) platforms, requires the definition of a new strategy, considering the available hardware/software alternatives.

This paper analyses the experience of the EPS@ISEP students regarding the use of microcontroller based platforms in the development of engineering capstone projects, and proposes possible future hardware/software alternatives, both from the technical and pedagogical perspectives.

Keywords: Microcontrollers, Project-Based Learning, Programming, STEM, Engineering Education.

1 INTRODUCTION

During the EPS@ISEP projects, multicultural and multidisciplinary groups of students face the challenge of solving open ended problems, with a particular emphasis on sustainability related projects. Many of those projects need some degree of automation, that many times is implemented through a microcontroller [3].

As one of the objectives of EPS@ISEP is fostering the teamwork, it is desirable to involve in the microcontroller related tasks the maximum number possible of students. This implies to give formation in electronics and embedded programming to students from very diverse backgrounds [4], in a preparatory “Technological Crash Course”.

One of the constraints when choosing a microcontroller board is pedagogical. There should be documentation available and in a format suitable for beginners. One of the aspects that is also pedagogical, or should be considered as pedagogical, is the type of difficulties encountered when developing the project. For instance, using an already made board avoids the need to design a PCB (Printed Circuit Board), and using a bought power supply avoids the need to build one. This allows the students to concentrate their efforts on the most important tasks and should not be seen as easing the student's work, but instead concentrating the difficulties on the most important areas of the work, in order to focus the student's attention on those areas.

So, the choice of microcontroller board, has to be made considering the pedagogical implications of the possible technological (or practical) difficulties faced by the students when developing their work.

Unless they are scavenged from previous projects, the parts need to be bought. This presents two difficulties: getting the money and getting permission to buy. Sadly, students have the attitude "It seems nice, we will buy it!", that results from many impulsive buys in their private life.

First, there is the elaboration of a list of parts, then the approval, and then the parts are bought. This usually introduces some waiting time in the project but this is positive because:

- Students are forced to specify which concrete (not abstract) parts are needed.
- Students must take in consideration a finite budget.
- Students learn to organize shopping lists to minimize the number of suppliers.
- Students are forced to commit into writing their design decisions.

2 DEVELOPMENT BOARDS

2.1 Arduino

The Arduino family of microcontroller boards, with the Arduino Uno as the most know example, has provoked in revolution in the development of embedded projects, lowering the technical obstacles to their implementation [5].

The Arduino offers a simple and flexible architecture[6], allied to a simplified programming environment, that makes possible a fast feedback cycle, and a soft learning curve, even for beginners. Their success is a proof of their ease of use, but also has given origin to many internet sites with wrong of misleading information about technical details.

The EPS@ISEP program believes in learner autonomy and tries to give it a proper place in the learning process, but after some analysis it was decided to provide the EPS students with a "Crash Course" on Electronics and Arduino, to give them some initial guidance in the learning process.

Different Arduino boards (Uno, Mini, Nano, Mega) have been used successfully since the first edition of the EPS@ISEP (in 2011) and the experience with their use has been positive. Meanwhile, the need for more processing power in some projects, and more sophisticated connectivity options has motivated the search for other options.

2.2 STM32

The STM32 family of processors from ST, has a low price, has 32 bit registers (8 bits on the *normal* Arduino) an higher clock frequency, and lower power consumption than the Arduino. One of cheapest boards available is the so called "Blue Pill" board based on the STM32F103C8 processor. On eBay and Aliexpress it can be bought at around 2 Euros and the low price and good performance make it a very interesting board. A group of users has created a port of the Arduino software for it (http://www.stm32duino.com). Sadly, the quality control on the boards is lacking, the boards usually need an additional JTAG interface for programming and are not very beginner friendly. ST has a series of boards called ST Nucleo with a wide range of processors and a very good build quality, but more expensive and only some of them have STM32duino support.

These boards are recommended for prototyping embedded systems but are not suitable for use by a beginner. Someone who is still learning embedded systems, might get lost not only in the difficulties, but also in the multiple possibilities of configuration and use of the STM32 processor. Up to date no team has used this kind of boards.

2.3 TI Launchpads

Texas Instruments has a series of low cost, low power boards with processors that include the 16 bit MSP430 processor variants and the 32 bit TM4C processors. Their cost is low when compared to an Arduino, their performance and power consumption are also very good and they have a series of "booster packs", that are plug-in boards that facilitate system integration. Besides having an Arduino like software called Energia (http://energia.nu), the available documentation is excellent, and in case of advanced use all the boards support JTAG debugging (unlike the Arduino).

One of the EPS@ISEP projects has utilized this board with very good results [7].

2.4 ESP8266

The ESP8266 processor from Espressif started as a co-processor designed to provide a Wi-Fi interface for other processors like the Arduino processor, through a modem-like serial interface. As the processor used has a greater performance than the AVR processor used on the Arduino, using an higher performance processor to offload some tasks from a low performance processor does not make sense. So, some boards were designed like the Node-MCU (http://www.nodemcu.com/) where the ESP8266 is the only processor, to lower costs, provide a higher performance and simplify the software development.

2.5 ESP32

The ESP32 aims to be an improved version of the ESP8266, having not only Wi-Fi, but also an additional Bluetooth interface, that widens the range of possible applications. Another thing to consider is that systems designed around the ESP32 can reach a lower overall power consumption than those based on the ESP8266. On Table 1 there is a summary table considering various embedded boards.

Table 1. Embedded boards comparison.

	Arduino Uno	"Blue Pill" STM	MSP430	TM4C	ESP8266	ESP32
Processor	AVR Atmega	STM32F103C8	MSP430	TM4C	ESP8266	ESP32
Clock (MHz)	16	72	16	80	80	160
Proc. Bits	8	32	16	32	32	32
ROM (kB)	32	128	16	256	≥ 512	≥ 512
RAM (kB)	2	20	0.5	32	160	512
ADC (bits)	10	12	10	2*12	10	12
Wi-Fi	No	No	No	No	Yes	Yes
Bluetooth	No	No	No	No	No	4.2

3 DEVELOPMENT LANGUAGES

As the processors used in low cost boards have gained power, other alternatives to "Arduino-like" programming started to appear. The Arduino software development environment built around a Java IDE (Integrated Development Environment) that uses a C++/C compiler underneath, with a series of libraries that hide the complexity of programming an embedded processor. While this may be friendly for a beginner, sometimes the details are not hidden in the right way or may not be considered by the programmer. Considering for instance, that a program can be 20 times slower using a data type of float instead of integer, this can be a critical factor in some programs.

Another factor to be considered is the progressive integration of embedded boards in complex IoT systems. This implies a wider programming perspective, and an increased sophistication of the libraries used. This can be seen in the recent adoption of high level languages for embedded systems, like Lua (http://www.nodemcu.com/), JavaScript (https://www.espruino.com) or even Python (https://micropython.org) [8].

4 CONCLUSIONS

While the majority of the EPS@ISEP projects has used the Arduino boards up to now, and in a successful way, the change of platform and of development language is being considered. The ESP32 based boards are a good compromise between performance and low power consumption, and already have Wi-Fi and Bluetooth interfaces built-in.

From the programming languages cited above, the use of a variant of Python for Microcontrollers will allow to use of a single language for various uses [9]. Prototyping algorithms, task automation, web programming, embedded programming and data processing are tasks that can be made with the Python language.

The change to the ESP32 processor has started this year, with two projects, and the change to MicroPython may happen next year.

REFERENCES

[1] M. F. Silva, B. Malheiro, P. B. Guedes, P. D. Ferreira, and A. Duarte, ‘The European Project Semester at ISEP (EPS@ISEP) programme: Implementation results and ideas for improvement’, in *Proceedings of the 45th SEFI Annual Conference 2017 - Education Excellence for Sustainability, SEFI 2017*, Azores, Portugal, 2017, pp. 129–130.

[2] A. Dziomdziora, D. N. Sin, F. Robertson, M. Mänysalo, N. Pattiselano, A. Duarte, B. Malheiro, C. Ribeiro, F. Ferreira, M. F. Silva, P. Ferreira, and P. Guedes, ‘Artistic Robot – An EPS@ISEP 2016 Project’, in *Interactive Collaborative Learning*, Springer International Publishing, 2017, pp. 225–238.

[3] A. Reinhardt, A. C. Esteban, J. Urbanska, M. McPhee, T. Greene, A. Duarte, B. Malheiro, C. Ribeiro, F. Ferreira, M. F. Silva, P. Ferreira, and P. Guedes, ‘Didactic Robotic Fish – An EPS@ISEP 2016 Project’, in *Interactive Collaborative Learning*, Springer International Publishing, 2017, pp. 239–253.

[4] A. Fountain, B. Kuron, C. Bentin, E. Davies, K. Suits, P. del Toro, A. Duarte, B. Malheiro, C. Ribeiro, F. Ferreira, L. Lima, P. Ferreira, and P. Guedes, ‘Learning sustainability with EPS@ISEP - development of an insectarium’, in *International Symposium on Project Approaches in Engineering Education – Vol 6*, Guimarães - Portugal, 2016, vol. 6, pp. 109–116.

[5] M. Margolis, *Arduino Cookbook*, 2nd ed. Sebastopol, CA: Oreilly & Associates Inc, 2012.

[6] D. Wheat, *Arduino Internals*. New York: Apress, 2011.

[7] A. Simons, J. Latko, J. Saltos, M. Gutscoven, R. Quinn, A. Duarte, B. Malheiro, C. Ribeiro, F. Ferreira, M. Silva, P. Ferreira, and P. Guedes, ‘Self-oriented solar mirror - An EPS@ISEP 2017 project’, in *TEEM 2017 Proceedings of the 5th International Conference on Technological Ecosystems for Enhancing Multiculturality*, Cadiz, Spain, 2017.

[8] C. Bell, *MicroPython for the Internet of Things*. New York, NY: Apress, 2017.

[9] N. H. Tollervey, *Programming with MicroPython*. Sebastopol, CA: O’Reilly Media, Inc., 2017.