International Journal of Engineering Education Vol. 35, No. 2, pp. 510–518, 2019
Printed in Great Britain

0949-149X/91 $3.00+0.00

An Interdisciplinary Approach to Motivate Students to Learn Digital Systems and Computing Engineering*

MARTA I. TARRÉS-PUERTAS, ALEXIS LÓPEZ-RIERA, PERE PALÀ-SCHÖNWÄLDER and SEBASTIÀ VILA-MARTA
Department of Mining, Industrial and ICT Engineering (EMIT),Universitat Politècnica de Catalunya, Spain.
E-mail: marta.isabel.tarres@upc.edu, alexis.lopez.riera@upc.edu, pere.pala@upc.edu, sebastia.vila@upc.edu

We report a new learning approach in collaborative learning-by doing, real-world team-based project in two ICT courses: Digital Systems and Computing Engineering, conducted at Universitat Politècnica de Catalunya. Data collected included: background information on students; course evaluations; measures of the knowledge and cross-knowledge of both disciplines taken before and after our SimulAVR project. SimulAVR integrates interdisciplinary knowledge by simulating via software a microcontroller and its implementation in VHDL. Our study is based on the analysis of the results of running the project for 3 years. After taking the simulAVR project, the students rated the interest in both courses higher.

Keywords: interdisciplinary projects; learning by doing; problem-based learning; collaborative work; teamwork; multidisciplinary approach; ICT; microcontrollers; simulation; digital systems; motivation; computing engineering

1. Introduction

Empirical data from a qualitative investigation of the experiences of ICT students cursing the Bachelor's degree in ICT Systems Engineering at the Universitat Politècnica de Catalunya (UPC) is presented. The new Bachelor's degree in ICT Systems Engineering was born on September 2010 as a response to the adoption of the European Higher Education Area (EHEA). Bachelor's degree in ICT Systems Engineering trains and qualifies students to deal with engineering problems in which electronics, informatics and communications play a part. ICT Students have a general training profile and the specific knowledge required to work in the emerging, high-impact sector of embedded systems, which are present in automotive, home automation, industrial machinery, medical equipment, consumer electronics and traffic control systems. In this context, two new subjects called Programming Technologies (PT) and Digital Systems (DS) were designed in the second semester of the first year.

The relevance of PT in the Bachelor is paramount and students learn the object oriented programming paradigm in Python. The standard computer science discipline curricula recommendations, ACM and IEEE-CS, (see for example [1]), emphasize the relevance of learning new languages and programming constructs, and suggest that students must understand the principles underlying how programming language features are defined, composed, and implemented. One of the goals of PT is to show students that building mathematical models of real systems can help in the design and analysis of those systems; and to encourage students with the difficult step of deciding which aspects of the ICT world are important to the problem being solved and how to model them in ways that give insight into the problem.

The aim of DS is to learn how to design complex digital systems. The programmable devices (FPGAs, CPLDs) allow testing the design of digital systems implementations described by VHSIC Hardware Description Language, in short, VHDL. VHDL language in introductory courses on digital systems is widely used. For more details on a VHDL language course, see for example, [2].

The objective of this paper is to describe our experiences, both successes and failures, in teaching PT and DS topics during three years and document what we have learnt, providing the details of the interdisciplinary project SimulAVR. This paper describes the use of an AVR microcontroller (SimulAVR) in a project-based learning environment in which the design of the AVR is the basis of the PT project in order to encourage the students to get more involved with the computer digital systems abstraction. The students themselves take responsibility for their learning process and become promoters of their own learning. They have to develop a project that is relevant to the real world. They need to obtain the relevant information and learn all of the necessary concepts while the lecturer provides guidance and counseling in the process (they learn the concepts by doing). Thus, the project stimulates learning. Moreover the project is based on collaboration. Students have to work in teams, which forces them to develop other skills. For more information on the pedagogical approaches we refer to the works of learning by doing projects, [3, 4], problem-based learning, [5–7], project-based computer programming activities [8], collaborative

* Accepted 10 November 2018.

research between courses, [9–12], real-world projects and team-based learning, [13, 14]. All the course files for DS and PT are open source and available at the OpenCourseWare of this degree,(see [15]).

1.1 Motivation

PT and DS topics have been taught at universities using basically the same teaching methods for years, which include lectures to explain theoretical concepts and laboratory assignments to put them into practice, based on personal computers. In our approach we emphasize learning-by-doing in contrast to traditional lecture-based approach.

Programming Technologies course is a critical building block in information systems curricula, (see [16]). The fundamental concepts of OOP, including (but not limited to) writing classes, inheritance, and polymorphism, are taught. Python is used as the programming language, but the focus of the course is more on general concepts instead of features typical of any particular language. For previous work in this area see for example the works of Goldwasser and Letscher, [17].

Our work is significant because (1) we apply a new approach based on collaborative learning between digital systems and computing, (2) we apply novel pedagogical approaches such as problem-based learning, project-based computer activities, team-based learning and learning-by-doing project, (3) the approach allows improving student's learning evaluation in DS and PT compared to the former method where assignments were made individually for each course and (4) allows to keep the student's motivation and interest in DS and PT high.

One of the main challenges we have faced is to teach DS and PT in an interdisciplinary way that allows us to give high-quality feedback. We redesigned a programming course based on the best practices obtained from our own research and experiences. The redesign was done in the following areas: by enhancing active learning, collaboration and team-based learning, real-world projects and collaborative research between courses, and student communication and by refactoring the evaluation procedure. In this regard, we have observed how PT and DS teacher's enthusiasm and receptivity for developing the project SimulAVR based on digital systems is a critical ingredient for maximum impact. The SimulAVR project, [18], is a simulator for the Atmel AVR family of microcontrollers.

1.2 Main contributions

In this paper we present, (1) a multidisciplinary project, SimulAVR, using object oriented programming methodology learned in PT and implementation in VHDL, (2) the study of the satisfactory results obtained by combining concepts of the two disciplines: digital systems and computing, and (3) the lessons learned in setting up the SimulAVR project.

1.3 Methodology

Next follows the main stages of our methodology.

1.3.1 Sharing tasks and discussions by interdisciplinary teachers

To carry out this approach and be able to "train the trainer", the fellow colleagues of computer science and electrical engineering planned to meet one week before starting the course and weekly two hours during the semester. Theoretical study of the digital systems micro-controllers area are discussed with computer science teachers. In our case, it was really easy to break down barriers between team members of computer science and electrical engineering departments due to the fact that the departments were joined as a result of the new ICT bachelor.

1.3.2 Project proposal brainstorming

The objective was to develop a preliminary design of the new software project challenge as a follow up of the works already developed in PT and DS.

1.3.3 Design the object-oriented paradigm of the SimulAVR project and verification of the algorithm resulting as the proposal solution

This is worked out together in face-to-face meetings with agendas determined by with DS and PT teachers. The project is broken into tasks and classroom teaching is synchronized in order to prepare students for each task of the project. These requires a strong collaboration between instructors before each guided task. We use a content management system (Subversion, SVN) to share source code and results. The project design included: (1) the development of the UML diagram, (2) specifications, (3) software project planning: requirements, core use cases, basic design, design verification, assignments to code in parallel, (4) test cases development, (5) initial integration by combining subcomponents/subclasses and (6) final integration. Finally, an efficient algorithm to simulate an AVR mini-controller is devised and tested. In section 3 we describe the SimulAVR project in depth.

1.3.4 Publishing material on-line [18] and reviewing material based on student's feedback

The published material includes (1) the SimulAVR specifications according to the software project planning, (2) Programming Technologies and Digital Systems course notes.

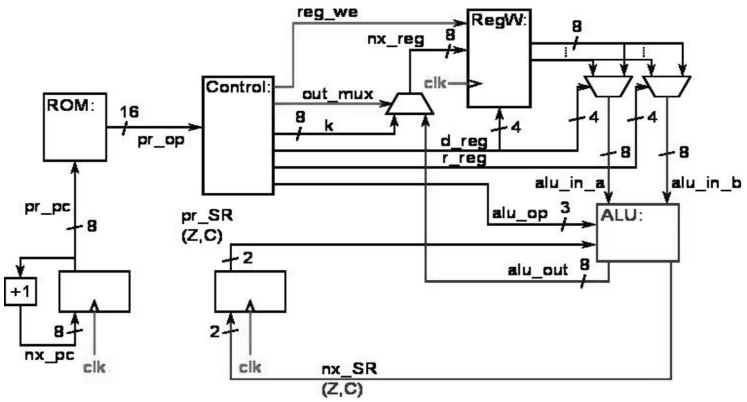

Fig. 1. MiniAVR architecture schematic (VHDL implementation).

1.3.5 Measuring the quantitative impact of our actuations

During the semester we gather two types of course data related to analyze the impact of the AVR project in PT and DS marks. We do this for two reasons: (1) to adjust the course to the students' needs during the course, and; (2) to analyze data after the courses in order to improve the performance of the courses the following year. In section 4.1 we give the details of the results.

1.3.6 Checking evaluation results as an analysis of student's marks and gathering feedback from students

We also want to use the class feedback and experience for improving the project statement, its distribution into tasks, adjusting the difficulty of each task, etc. Instructors spend considerable time supervising and interacting with the students. The Subversion management system gives us also the ability to closely monitor the progress of each group of students. A significant fraction of student groups are able to produce impressive solutions and most, if not all, are more than correct. And all of them provided very positive feedback for this task. In section 4.2 we give the details of the results.

1.4 Organization

The rest of the paper is organized as follows. Sections 2–3 give the required background and curricular context information about DS and PT courses, and the implementation details of the SimulAVR project. In Section 4, the evaluation data gathered and the results obtained each year are presented, followed by a description of the feedback from students, reflecting on the results and the adaptations made to during the three subsequent academic years in which it was applied, from 2013/2014 to 2015/2016. Section 5 discusses the lessons learned during this experience and draws conclusions.

2. Courses description

2.1 Description of digital systems

Digital Systems is a compulsory course in the second semester during the first year of the ICT degree. The main goal of the course is to design and implement digital systems in VHDL language. Thus, it is based on the design of small projects based on real devices, implementing them with a FPGA device. The concepts of digital systems theory used from now on are standard. For more information we refer to the books of Mano, [19], Tanenbaum [20] and Atmel Project [21].

On the first part of the course, basic concepts of digital systems applied to VHDL are introduced. On the second part, the aim is to reinforce the VHDL learning through the design and implementation of an 8-bit AVR microcontroller. The designed AVR is a simplified version from the original. The main VHDL code is given to the students. Fig. 1 shows the miniAVR architecture schematic.

The course evaluation consists of a partial exam and a final exam, that represent 15% and 35% respectively of the final mark. The partial exam contains some questions about general concepts in digital electronic and a VHDL problem. In the final exam there is a VHDL problem and a multiple choice test about specific questions on the AVR microcontroller (particularly on VHDL implementation) and featuring concepts on VHDL. Moreover, the lab assignments give 25% of the final mark and the exercises another 25%.

2.2 Description of programming technologies

Programming Technologies course is a critical building block in information systems curricula, (see [16]). The fundamental concepts of OOP are taught. Python is used as the programming language.

In the two hours of theory sessions, the professor

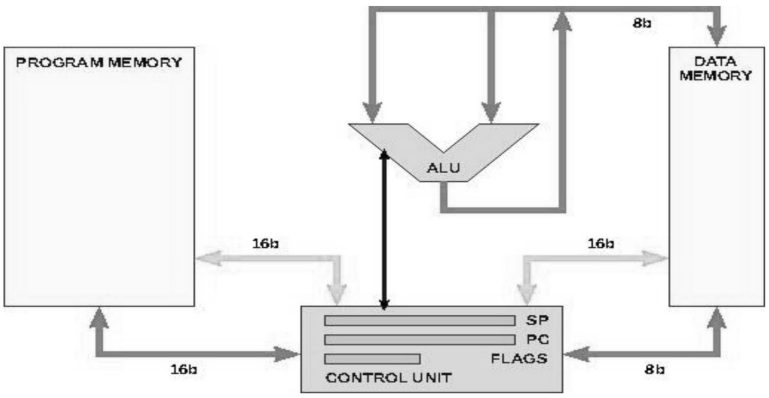

Fig. 2. SimulAVR microcontroller basic architecture.

alternates new theoretical concepts with examples and exercises. Lectures, in which the course topics are presented, explained and illustrated, are combined with student interaction regarding the various alternatives for solving practical cases. In the two hours of laboratory sessions, some exercises are solved through collaboration. To facilitate active learning, students are expected to solve problems from a list during laboratory sessions and as part of their home study activities. They are also advised to regularly consult their professor about the quality of their programs. Finally, students need to work in groups to solve the SimulAVR project. The programming project integrates knowledge and skills of the entire course. In a previous laboratory session, teams of students will be trained to solve the project. The final exam mark is awarded individually. Final marks for the course will be determined using the following weighting: 40% final examination, 25% SimulAVR project assessment, and 35% continuous assessment (laboratory and additional lecture exercises).

3. The SimulAVR project

In what follows we describe the new SimulAVR Python based project.

3.1 AVR microcontroller background

This section is a recapitulation of some basic concepts that are the base of the developed software. For more information on AVR 8-bit Instruction Set concepts we refer to [19], Tanenbaum, [20] and Atmel [21]. In what follows we provide the specification details of the AVR microcontroller architecture provided to PT students.

The AVR is a modified Harvard architecture 8-bit RISC single-chip microcontroller, where program and data are stored in separate physical memory systems that appear in different address spaces, but having the ability to read data items from program memory using special instructions. The AVR Enhanced RISC microcontroller supports powerful and efficient addressing modes for access to the Program memory (Flash) and Data memory (SRAM, Register file, I/O Memory, and Extended I/O Memory). This section describes the various addressing modes supported by the AVR architecture.

Program instructions (ProgramMemory) are stored in non-volatile flash memory. Although the MCUs are 8-bit, each instruction takes one or two 16-bit words. The data address space (Data-Memory) consists of the register file, I/O registers, and SRAM. The working registers are mapped in as the first 32 memory addresses (0000–001F). Central in the AVR architecture is the fast-access RISC register file, which consists of 32 $\times$ 8-bit general purpose working registers. Within one single clock cycle, AVR can feed two arbitrary registers from the register file to the ALU, do a requested operation, and write back the result to an arbitrary register. The ALU supports arithmetic and logic functions between registers or between a register and a constant. Single register operations are also executed in the ALU. In addition to the general registers, there is a Program Counter (PC) register and a Status Register (SREG) that contains the condition flags and interrupt masks. The width(size) of the program counter (PC) is 16 bits.

Fig. 2 shows the basic architecture of the AVR microcontroller.

In the project a limited set of AVR instructions and a basic number of addressing modes is used. Following there is a description of the expected addressing modes needed by the instructions.

- Direct Single Register Addressing. The operand is contained in register d (Rd).
- Direct Register Addressing, Two Registers. Operands are contained in register (Rr) and (Rd). The result is stored in register (Rd).
- I/O Direct Addressing. Operand address is contained in six bits of the instruction word.

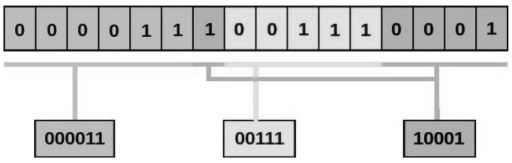

Fig. 3. SimulAVR Instruction ADD R7, R17.

The set of instructions needed to implement are the basic and arithmetic instructions, data transfer instructions, branch instructions and MCU Control Instructions. The set of required instructions are ADD, SUB, SUBI, AND, OR, EOR, LSR, MOV, LDI, STS, LDS, RJMP, BRBS, BRBC, NOP, BREAK, IN, OUT. For the case of IN and OUT instructions, we work with virtual ports, such as the keyboard and the screen, and we use the instructions for the in/out of the assembly language programs.

Fig. 3 shows the use of the AVR ADD instruction.

3.2 SimulAVR module structure

This section illustrates the overall architecture of the developed software, presents the major class designs and highlights application of Python language for simulating the miniAVR. The project is structured in several modules containing one ore more classes. Following there is a brief description of each module.

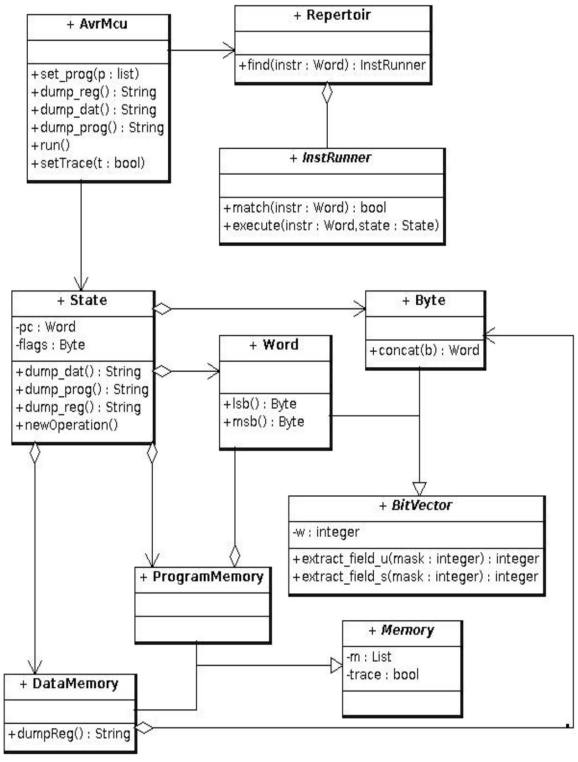

Fig. 4. SimulAVR class diagram.

bitvec: Contains the classes needed to represent words and bytes.

memory: Contains the classes related to the memory of the AVR architecture.

state: Contains one class that represents the state (memory included) of the microcontroller.

instruction: Contains the classes that implement all and each of the operations in machine language recognized by the simulator.

repertoir: Contains one class that groups the repertoir of instructions of the simulator.

avrmcu: Contains one class that implements the general control of the microcontroller.

intelhex: It is a predefined module that allows to read conveniently programs written in machine language, that is files with Intel HEX format, [Int88].

avrexcep: This is the module needed to manage the exceptions.

simavr: This is the main module.

3.3 SimulAVR class diagram

Fig. 4 shows the class diagram that represents the class structure of the project. Note that neither constructors nor special methods are included. Following there is a brief description of each class.

BitVector class is an abstract class representing a word less than or equal to 16 bits.

Byte class represents a byte.

Word class represents a word of 16 bits.

AVRException class handles any exception raised in the simulation of the AVR. AVRExceptions are classified into BreakException, MemoryException and UnknownCodeError.

Memory class is an abstract class and represents the space memory of the AVR. It has two subclasses, the ProgramMemory class and DataMemory class. Data memory is addressed by octet. Load and store instructions operate on 8 bit quantities. Program memory is addressed by 16 bit word when fetching instructions.

State class represents the MCU state. The state contains the set of registers and memories. When an instruction is executed, the state changes.

InstRunner class is an abstract parent class which generalizes the common methods for each of the instruction subclasses. Each instruction of the microcontroller is represented by a class. For example, the ADD instruction is represented by the Add class, and similarly for each of the rest of instructions implemented for the AVR. The classes Adc, Sub, Subi, And, Or, Eor, Lsr, Mov, Ldi, Sts, Lds, Rjmp, Brbs, Brbc, Nop, Break, In, Out are subclasses of InstRunner.

Repertoir class represents a set of instructions of a

MCU. The main goal of this class is, given an instruction returns the InstRunner object that is capable of executing the corresponding code.

AvrMcu class represents the MCU AVR simulator. The main goal of this class is to execute a program written in assembly language.

3.4 SimulAVR program execution

The main program is invoked as follows, *$ ./simavr program.hex*

The file program.hex contains the program for the MCU AVR in the standard format HEX, [22]. When finishing the execution of the simulation, command simavr, allows to do a set of actions according to the following options:

-p: the program memory is dumped to screen.
-r: the registers content is dumped to screen.
-d: the data memory content is dumped to screen.
-t: set trace on the data memory operations.

For example, *$ ./simavr -rt program.hex* executes the instructions contained in the *program.hex* file, shows the data memory accesses and dumps registers state. The file in HEX format contains the machine program needed by the AVR. IntelHex project, [23], gives a python module that allows to easily process files in HEX format. Class *IntelHex*16bit from *intelhex* module is used to read the machine language instructions stored in he HEX file that needs to be simulated. In order to create de HEX file we use a free assembler language for the AVR family, named Avra, [24]. Then we can write the file example1.asm and then convert it automatically to example1.hex.

```
LDI R16, 0xff
LDI R17, 1
ADD R17, R17
MOV R0, R17
AND R0, R16
BREAK
```

The avra commands are as follows: *$ avra example1.asm* and *$avra -l example1.lst example1.asm*

4. Evaluation results

A total of 120 students participated in this study, with an average of 40 students each course. The students were randomly assigned to collaborative learning teams, each of which consisted of four students. The core of the experiment consisted of an 8-week software development project based on the AVR micro-controller simulation. At the beginning of the learning activity, the students were asked to understand the UML diagram given in object oriented classes together with an accurate description of methods and attributes of each class. Following that, they needed to design and implement the software for simulating a small AVR-compatible microcontroller (SimulAVR). In our research, we analyzed (1) student's marks in DS and PT courses, (2) student's perception of their ability to solve digital systems problems using PT, and (3) student's learning attitudes toward taking DS and PT courses, compaired to other courses in the institution. The aim of the study was twofold: to assess the learning effectiveness of the new active educational approach and to analyze the motivation toward the two subjects fostered by the project development. The first aim is analyzed using the scores achieved by the students in each course; the second aim is analyzed via satisfaction questionnaires. We revisit these points in more detail in the next subsections.

4.1 Analysis of students' marks

Next follows the analysis of student's marks in PT and DS through the effects of the SimulAVR project implementation. The official marks provided by UPC-Manresa were employed to compare the students' problem-solving ability after solving the SimulAVR assignment in each course (DS and PT), compared to the ones before doing the assignment. Fig. 5 shows the total scores of the students compared for the periods before (Pre-AVR) the SimulAVR project development (from academic year 2013/14 to academic year 2015/16) and after the finalization (Post-AVR). In each course we can see an increasing of the student average marks in each subject after doing the assignment (Post-AVR) with respect to the Pre-AVR marks. For the case of DS, the marks augmented by 27%, 24% and 36% respectively. For the case of PT the marks increased 5%, 10% and 4% respectively.

4.2 Gathering feedback from students

In what follows we provide the information about students viewpoint related to the ability to solve DS and PT problems, including students fulfillment perception measures.

4.2.1 Analysis of students' problem-solving ability and Analysis of the students' learning motivation

The same questionnaire was given to students at the middle of the semester (Pre-AVR) and at the end (Post-AVR). Students ranked their level of agreement or disagreement (1: Strongly disagree, 5: Strongly agree) with several predefined statements. Table 1 lists the eleven statements related to the project activities and their feeling of level of understanding of the concepts of the two courses.

Fig. 6 indicate a strong perception of students to their own ability to work in Object-oriented assignments based on Digital Systems.

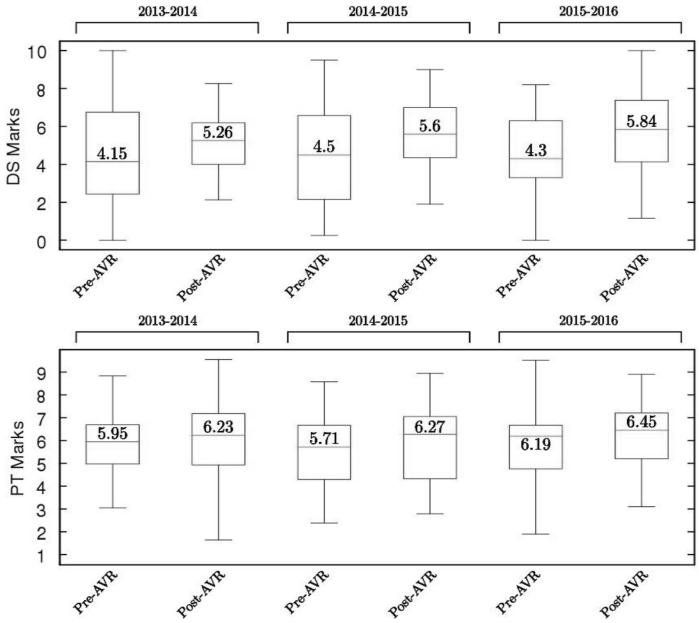

Fig. 5. Analysis of students' marks in PT and DS.

Table 1. Specific questionnaire

Q1: VHDL language is a powerful language used widely.
Q2: I am able to apply the VHDL language in order to solve any problem presented in DS field.
Q3: I am able to apply the OOP concepts learned in the PT course to real issues or to other disciplines.
Q4: DS practical cases are widely applied in the industry.
Q5: Concepts in DS and in PT match together.
Q6: PT allows to deepen in the knowledge of the AVR working process.
Q7: PT increases the knowledge of the OOP methodology.
Q8: PT allows to practice in depth the VHDL language.
Q9: PT helps to understand the relationship between high level and low level programming languages.
Q10: PT helps to simulate how the miniAVR works by using a limited set of instructions.
Q11: The AVR concepts in DS are the base of the SimulAVR

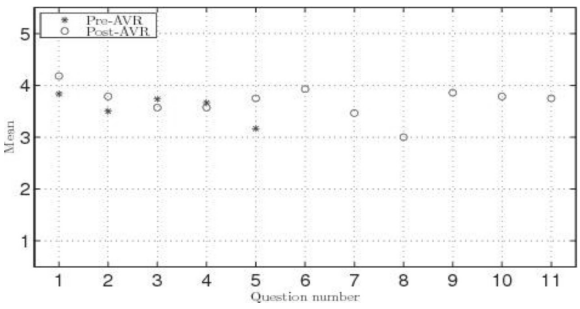

Fig. 6. Specific questionnaire (Table 1) results.

4.2.2 Analysis of students' learning attitudes toward taking the courses

Finally we provide the analysis of students' learning attitudes toward taking PT/DS courses compared to all courses in UPC and all courses in UPC-Manresa (EPSEM). Table 2 shows the general questionnaire from official evaluation results carried by UPC-BarcelonaTech at the end of the semester.

Table 2. General questionnaire at UPC

Q1: The contents of the course are interesting.
Q2: I agree with the course procedures and assignments support course objectives.
Q3: I am satisfied with the course

The students are asked to mark the level of agreement in each statement of Table 2 from 1 (Strongly disagree) to 5 (Strongly Agree). The satisfaction and total scores of the students were compared from academic year 2013/14 to academic year 2015/16. Fig. 7 shows, for each question, the mean of ranks obtained in the three academic courses.

The results in Fig. 7 indicate that the intervention

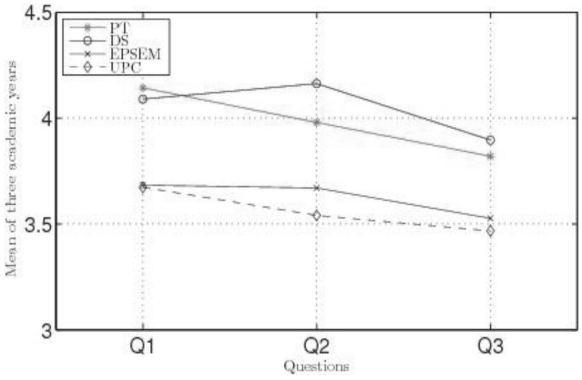

Fig. 7. Results of general questionnaire at UPC (Table 2).

of the proposed learning approach improves the learning motivation of the students in the two courses. Further, the interest in DS and PT is higher than the median of the courses taken in UPC as a whole and UPC-Manresa (EPSEM). However, in order to be able to contextualize these numbers and emphasize the success of our proposal, note that, in our institution, the passing rate in the second semester of first-year subjects is around 38% and the drop-out rate around 44%, both on average, (see [25]), while the numbers corresponding to DS and PT (the full subject) are around a 80% passing rate and 10% drop-out rate.

5. Discussion

We have found that in each group there is a number of students who enjoy challenges, and thus like the challenging projects as the SimulAVR project as well. Student's attitude is very important for a successful SimulAVR project. Also, our approach has some limitations in the case of students that fail one of the two courses because it becomes very difficult for them to be able to reach the challenge of relating and integrating the knowledge of both subjects. The percentage of these students is low, and the number of students who leave both courses almost always coincides.

6. Conclusions

The SimulAVR project allows ICT Systems Engineering students to deepen their knowledge of digital systems together with object oriented programming. Empirical evidences developed for both courses at UPC in Manresa have been presented. Through these activities, students became more interested in the Digital Systems and OOP concepts. Additionally, the number of students achieving high scores increased. What is more, there is an increment of the student interest in taking computer architecture subjects in the subsequent academic year. The positive impact of this experience on teaching in first-year subjects suggests that this idea can be expanded to other courses. Student feedback and assessment data indicate that the intended objectives were achieved.

It is hoped that the findings of this study can assist the lecturers of OOP and DS courses to develop similar techniques to improve student learning experiences and motivate students towards Digital-Systems-OOP and related courses. All exercises, data and supplementary material are available at openCourseWare web-site of this degree, for use by other instructors. The four authors of this paper won the ''Premi a la Qualitat en la Docència Universitària del Consell Social de la UPC al Grau en Enginyeria de Sistemes TIC'' Award, which was awarded annually for excellence and innovation in teaching and for making a significant impact on the student learning experience.

Acknowledgments—This study is supported in part by the Spanish Ministerio de Economia y Competitividad and FEDER grant TEC2015-65748-R. The authors would like extend our thanks to DS and PT students who provided excellent feedback over the past 3 years.

References

1. IEEE Computer Society The Association for Computing Machinery (ACM). Computer Science Curricula 2013. Curriculum guidelines for undergraduate degree programs in computer science, https://www.acm.org/education/CS2013-final-report.pdf, Accessed 1 June 2017.
2. R. d'Amore, A Synthesis-Oriented VHDL Course *ACM Transactions on Computing Education*, **10**(2), pp. 1–24, 2010.
3. M. V. Ramakrishna, A Learning by Doing Model for Teaching Advanced Databases, *Proceedings of the Australasian Conference on Computing Education, ACSE '00*, New York, NY, USA, pp. 203–207, 2000.
4. A. V. Alejos, J. A. G. Fernandez, M. G. Sanchez and I. Cuiñas, Innovative Experimental Approach of Learning-Through-Play Theory in Electrical Engineering, *The International Journal of Engineering Education*, **27**(3), pp. 535–549, 2011.
5. D. Chinn and K. Martin, Collaborative, Problem-based Learning in Computer Science, *Journal of Computer Science in Colleges*, **21**(1), pp. 239–245, 2005.
6. M. Cirstea, Problem-Based Learning (PBL) in Microelectronics, *The International Journal of Engineering Education*, **19**(5), pp. 738–741, 2003.
7. W. Hamiza, W. M. Zin, A. Williams and W. Sher, Introducing PBL in Engineering Education: Challenges Lecturers and Students Confront, *The International Journal of Engineering Education*, **33**(3), pp. 974–983, 2017.
8. H. Wang, I. Huang, and G. Hwang, Comparison of the effects of project-based computer programming activities between mathematics-gifted students and average students, *Journal of Computers in Education*, **3**(1), pp. 33–45, 2016.
9. E. L. Tiessen and D. R. Ward, Developing a Technology of Use for Collaborative Project-based Learning, *Proceedings of the 1999 Conference on Computer Support for Collaborative Learning, CSCL '99*, Palo Alto, California, 1999.
10. L. C. Benson, M. K. Orr, Sherrill B. Biggers, W. F. Moss, M. W. Ohland and S. D. Schiff, Student-Centered Active, Cooperative Learning in Engineering, *The International Journal of Engineering Education*, **26**(5), pp. 1097–1110, 2010.
11. J. J. Marquez, M. L. Martinez, G. Romero and J. M. Perez, New Methodology for Integrating Teams into Multidisciplinary Project Based Learning, *The International Journal of Engineering Education*, **27**(4), pp. 746–756, 2011.
12. H. C. Huang, S. G. Shih and W. C. Lai, Cooperative Learning in Engineering Education: a Game Theory-Based Approach, *The International Journal of Engineering Education*, **27**(4), pp. 875–884, 2011.
13. J. Tan and J. Phillips, Challenges of Real-world Projects in Team-based Courses, *Journal of Computers in Education*, **19**(2), pp. 265–277, 2003.
14. Á. F. Blanco, D. Lerís, M. L. Sein-Echaluce and F. García, Monitoring Indicators for CTMTC: Comprehensive Training Model of the Teamwork Competence in Engineering Domain, *The International Journal of Engineering Education*, **27**(4), pp. 829–838, 2011.
15. UPC iTIC OpenCourseWare, https://ocwitic.epsem.upc.edu, Accessed 1 June 2017.
16. Programming Technologies course notes, https://ocwitic.epsem.upc.edu/assignatures/tecpro, Accessed 1 June 2017.
17. M. H. Goldwasser and D. Letscher, Teaching Object-oriented Programming in Python, *SIGCSE Bull*, **39**(3), pp. 365–366, 2007.

18. SimulAVR. MCU AVR Simulator, Programming Technologies course notes, https://ocwitic.epsem.upc.edu/assignatures/tecpro/laboratori-material/tecpro.-projecte-de-curs/, Accessed 1 June 2017.
19. M. Morris, *Computer Engineering: Hardware Design*, Prentice Hall, 1988.
20. T. A. Andrew and S. Tanenbaum, *Structured Computer Organization*, English, 6th edition, Pearson, 2012.
21. Atmel AVR 8- and 32-bit, http://www.atmel.com/products/avr, Accessed 1 June 2017.
22. Intel. Hexadecimal Object File Format Specification, http://microsym.com/editor/assets/intelhex.pdf, Accessed 6 January 2017.
23. IntelHex Project, https://launchpad.net/intelhex, Accessed 1 June 2017.
24. AVRA—Assembler for the Atmel AVR microcontroller family, http://avra.sourceforge.net, Accessed 1 June 2017.
25. UPC Transparency indicators, https://gpaq.upc.edu/lldades/quadrecomandament.asp?codiCentre=330, Accessed 1 June 2017.

Marta I. Tarrés-Puertas received the PhD degree in Software and the Master's degree in Computer Science Engineering from the Universitat Politècnica de Catalunya (UPC). Since 2005 she is a lecturer at the UPC. Her research and teaching activity is mainly related to computing, software design and innovation in learning methodologies.

Alexis López-Riera received the Eng. Tec. Telecommun. and Master in Electronic Eng. degrees from the Universitat Politècnica de Catalunya (UPC). He was a part-time assistant professor at the UPC (2010–2013). He is currently working towards his PhD in the area of super regenerative receivers and he is Professor of Digital Systems.

Pere Palà-Schönwälder received the Eng. Telecommun. and PhD degrees from the Universitat Politècnica de Catalunya (UPC). He is currently an Associate Professor at the UPC. He is the contact person of UPC's Communication Circuits and Systems Research group and has been the project leader of several government and industry-funded research projects.

Sebastià Vila-Marta is an associate professor of Computer Science at the Universitat Politècnica de Catalunya (UPC), Manresa, Catalonia. His research interests include solid modelling, geometric constraint solving, high level languages for geometric representations and graphic user interfaces for CAD.