Using PBL and Software Engineering technical skills to motivate Computer Science students to develop a microcontroller core: the 8051-like case study

Adriano da Cruz Sena
Policia Rodoviária Federal (SRPRF-BA)
Itabuna, Brazil
adriano.sena@prf.gov.br

Cristiano Souza de Alencar
Faculdade de Ilhéus (CESUPI)
Ilhéus, Brazil
cristianoalencaradv@gmail.com

Péricles de Lima Sobreira
Universidade Estadual de Santa Cruz (UESC)
Ilhéus, Brazil
plsobreira@uesc.br

Jauberth Weyl Abijaude
Universidade Estadual de Santa Cruz (UESC)
Ilhéus, Brazil
auberth@uesc.br

***Abstract*—A new market of electronic products has arisen due to a growing demand on portable and ergonomic embedded devices. The technology related to the development of such products allows the high integration of complex systems in only one chip. However, we can realize there is a discrepancy between the transistor integration capacity and the development processes of such systems, and this fact motivated the proposal of this work: a research developed by two Computer Science undergraduate students from the utilization of Project-Based Learning, as pedagogical approach, combined with computer languages/methodologies (UML, SystemC and ipPROCESS) in such a way to make the construction of embedded systems more fluid. The proposed framework was used to model an 8051-like microcontroller as case study.**

Keywords— Project-Based Learning, scaffolding, System-on-Chip, microcontroller, UML, SystemC, ipPROCESS

I. INTRODUCTION

Over the past decade, the world has witnessed the emergence of a market of electronic products capable of simultaneously providing information, entertainment and communication to its different consumers. The time-to-market requirement, combined with a market need to integrate so many features into single devices, have not matched the growing capacity to integrate multiple systems into only one chip (System-on-Chip, or SoC), thus creating a gap between the productivity of development and the capacity of integration allowed by new chip technologies [1].

Alongside this issue, it can be observed a large number of students from university computer degrees traditionally focused on development of software solutions, such as in Software Engineering and Computer Science programs, sometimes presenting difficulty and indifference to hardware courses. This situation might be justified by, for example: the limited time usually allocated for such subjects during computer undergraduate programs; the growing demand existing in the market of software applications, or even; the lack of opportunities to develop hardware projects, particularly in small and medium Brazilian urban centers [2].

This reality was fundamental to introduce techniques which could motivate two Computer Science research initiation students (in the last academic year) at Universidade Estadual de Santa Cruz (UESC) in the design of an 8051-like microcontroller [3], from a framework composed by pedagogical techniques (PBL – Project-Based Learning) and high-level computer languages/methodologies (combining UML, SystemC and ipPROCESS).

For this purpose, this article is organized as follows: section 2 presents the advantages of using PBL as a possible technique to motivate students to learn; section 3 describes some languages/methodologies/processes used in the development of hardware projects from the application of concepts widely diffused in the domain of software engineering; section 4 presents our project framework; section 5 describes the process used to map UML diagrams in SystemC code, and finally; section 6 ends this article with conclusions and some ideas to be carried out in future works.

II. PBL AS A LEARNING APPROACH

The contemporary society has undergone remarkable changes in the last few decades due to the progress made in areas of scientific research and technological innovation. In this context, the higher education did not only have to adapt it to the needs of the knowledge society, but also had to play a main role in this process from the discussion of new concepts of productivity, cooperation and competitiveness.

This reality, and the lack of interest of UESC Computer Science students with regard to courses in the hardware domain, encouraged us to work during one academic year in a team composed by two scholarship undergraduate students supervised by two full researchers, all of such program (Computer Science), in a project scaffolded by PBL in the field of Computer Engineering.

The PBL approach is focused on the student (student-centered learning). Teachers act as facilitators, encouraging and giving sufficient guidance and feedback to their students. They highlight questions issued by professional and ethical goals while working on projects [4], having the chance to observe the abilities and difficulties of their students, who

DOI 10.14684/EDUNINE.1.Vol.2.2017.6-10

participate as designers of their own knowledge and in accordance with their rhythm, being free to choose strategies to be applied in the solution of real-life situations.

Students working on this technique should be considered “self-managers”, finding sources, conducting research and holding their colleagues responsible for their activities completion. They are encouraged to explore topics of their interest and to acquire self-assessment, communication, planning, collaboration and problem-solving skills, strengthening their confidence and self-esteem [5].

In our research, in particular, PBL guidelines were applied during all project schedule, where the two concerned students were required to make their own decisions through, for example, the division of labor (specializing, breaking down and distributing large activities into many tiny tasks) and the identification of alternative solutions to problems found (e.g., optimizing a loop in a method in order to satisfy real-time constraints).

These students played at different moments opposite roles, always in a collaborative way (a same role was played by both students in a rotating mechanism): “questioner/answerer”, in the Literature Review phase; “requirements specifier/ requirements reviewer”, “designer/design reviewer” and “implementer/test designer”, in the Development phase, and; “writer/reader”, in the Completion phase (this mechanism will be presented in details in section 4).

III. Software Development Methodologies/Processes Used in the Design of Embedded Systems

Due to the complex nature of embedded systems, the risks associated with the success of their developments already arise from the specification of their requirements. This fact, combined with the possibility to develop hardware projects from object-oriented programming languages (e.g. SystemC, a C++ library with class and macro definitions), allowed SoC projects to reuse techniques and development processes already extensively discussed and used in the Software Engineering field [6].

In hardware description languages (e.g. VHDL), only abstractions in the structural domain can be expressed. SystemC, in turn, supports the integration of multiple heterogeneous modules and has the ability to work at high levels of abstraction through the concept of channels, and behavioral modeling of systems from the use of ordinary classes in C++ [7].

The principles of abstraction and separation of concerns in object-oriented programming are expressed through the concepts of classes, attributes and methods, allowing the notion of object encapsulation that limits external access to private members. SystemC also allows mechanisms where an object can be instantiated from a subclass of a declared type [8].

The Unified Modeling Language (UML) is a visual language used to model software applications from the use of graphical notations. However, the use of SystemC in the development of hardware projects has compelled the unification of these (until recently) distant worlds [9]: UML can be used with object-oriented hardware description languages as SystemC in order to formalize the specification of embedded systems in high levels of abstraction. SystemC has a great similarity with UML-RT (UML for Real-Time systems): modules in SystemC correspond to capsules in UML-RT, where the former communicates via ports and channels, and the latter, via ports and protocols [1].

Therefore, the use of UML in the design of SoC systems is growing rapidly due especially to the following advantages offered by this language [10]: UML lets us graphically specify a system to a better understanding about the functional specification of the SoC to be developed; UML helps programmers to model behaviors, functionalities and architectures from the use of visual diagrams, automatically generating test scenarios for future implementation, and; UML has well-defined notations to describe specifications in a more concise way, avoiding possible ambiguities usually allowed by natural languages.

There are currently some proposals offering automated mapping from UML models to SystemC code (e.g. [11], [12]), where C++ classes are automatically constructed from UML Class and State diagrams. In our research, we performed this mapping in a semi-automated way, where the elaboration of other UML diagrams (Use Case, Sequence and Collaboration) was necessary for a better understanding of our case study design.

Finally, in order to provide a disciplined way to assign tasks and responsibilities in the design of our project, we chose ipPROCESS as its development process [13]. Although ipPROCESS has been created to be used in the development of embedded hardware projects to be prototyped on field-programmable logic devices (FPGA), this framework can be adapted to our objectives if we disregard one of its workflows (the “FPGA Implementation” discipline) and one of its phases (the “FPGA Prototyping” phase), from its general process (Fig. 1) [1] (the use of ipPROCESS in our framework will be presented in details in sections 4 and 5).

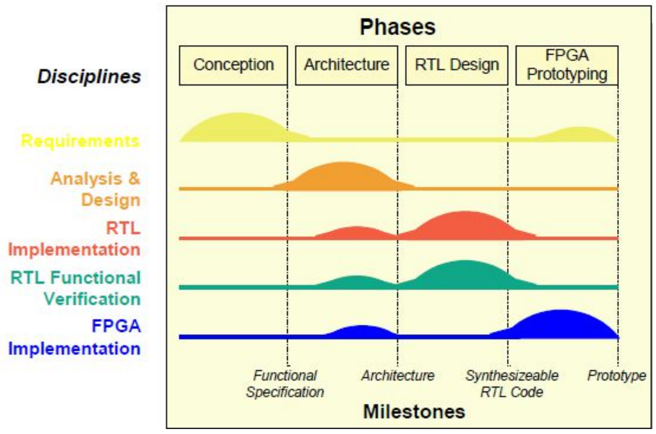
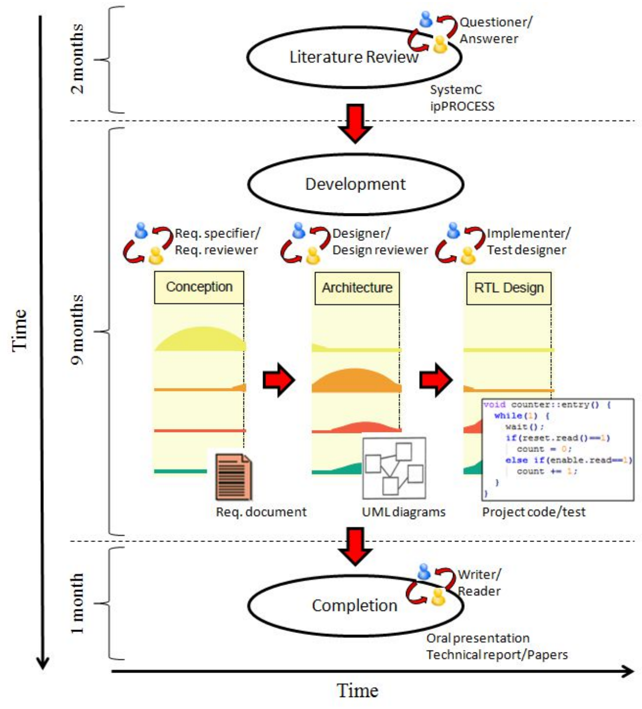


Fig. 1. ipPROCESS general vision (Disciplines *vs.* Phases)

IV. Framework

As discussed in later sections, this one-year scientific project was developed by two computer science undergraduate students working in the computer engineering domain. Its schedule was composed by three main steps, subdivided into periods of 2-9-1 months (Fig. 2).



```
void counter::entry() {
  while(1) {
    wait();
    if(reset.read()==1)
      count = 0;
    else if(enable.read==1)
      count += 1;
  }
}
```

Fig. 2. Project framework

The framework description is presented as it follows:

1) Literature Review (2 months). In this first step the two students played collaborative roles (questioner/answerer) in such a way to learn topics not directly approached by their university programs:

a) SystemC (1 month). Activities: look for bibliographic material; study SystemC at RTL, functional and simulation levels; install the development environment (operating system, IDE, compiler, etc.); exercise SystemC from some small projects.

b) ipPROCESS (1 month). Activities: look for bibliographic material; study UML 2.0 and some of its profiles for embedded/real-time systems; study the process concepts (lifecycle, phases, workflows and artifacts); install a CASE tool to design UML diagrams.

2) Development (9 months). In this second step, supported by the ipPROCESS (see Fig.1), both students implemented the case study of this project (an 8051-like microcontroller). This step was subdivided into periods of 1-2-6 months, namely:

a) ipPROCESS Conception phase (1 month). In this phase the students were responsible to elicit the requirements of our case study. Roles played: requirements specifier/ requirements reviewer. Activity: write the requirements document.

b) ipPROCESS Architecture phase (2 months). In this phase the students modeled the project in UML to represent, from some diagrams, as the microcontroller source code would be structured and written. Following the ipPROCESS, the students could implement and verify some project functionalities already in this phase, but we decided to execute these activities only in the next one. Roles played: designer/ design reviewer. Activity: create UML diagrams (Use Case, Activity, Sequence, Collaboration, State and Class diagrams).

c) ipPROCESS RTL Design phase (6 months). In this phase the students implemented and verified the microcontroller functionalities (in the next section we present the steps carried out to map UML models in SystemC code). Roles played: implementer/test designer. Activity: project implementation (including the construction of test cases and simulation files).

3) Completion (1 month). In this step the students performed final tasks. Roles played: writer/reader. Activities: write a final technical report; prepare the results to submit them into academic conferences, and; prepare an oral presentation to the annual undergraduate research week at UESC.

V. Mapping UML diagrams in SystemC code

This work was inspired by the case studies implemented in [14]-[16] from UML modeling of some 8051 microcontroller blocks, where interactions between this core and its environment were represented by Use Case diagrams. This UML representation contains Actors, representing external entities that interact with the system through well-defined interfaces, and Use Cases, representing system functionalities.

Fig.3 presents only a part of the Use Case diagram of our 8051-like microcontroller (the number of actors and use cases in this figure is minimal to keep it readable): Interrupt Control and Instruction Decoder modules. This diagram was elaborated using the Papyrus-RT platform, an Eclipse plugin that allows modeling Real-Time projects with automatic generation of C++ code from UML diagrams [17].

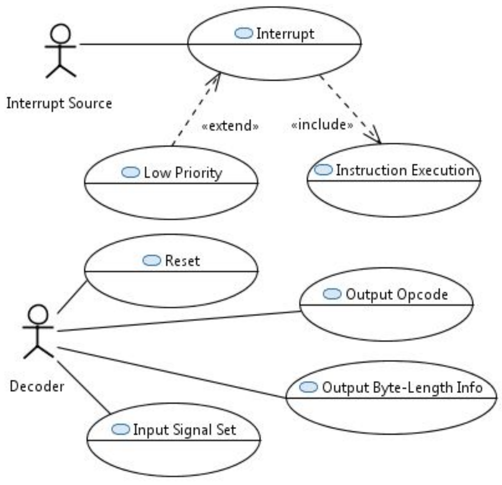


Fig. 3. 8051 UC Diagram for Interrupt Control and Decoder functionalities

The Use Case diagram helps us to define the classes of our project (Class diagram), and the Sequence and Collaboration diagrams, in turn, the behavior (methods definition) and relationship among these classes, respectively. Fig.4 presents as example the Decoder class, modeled in UML. As we can observe in this figure, the Decoder actor and its Use Case functionalities (Fig. 3) are mapped, respectively, on the Decoder Class name and its methods.

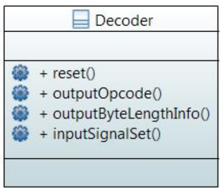
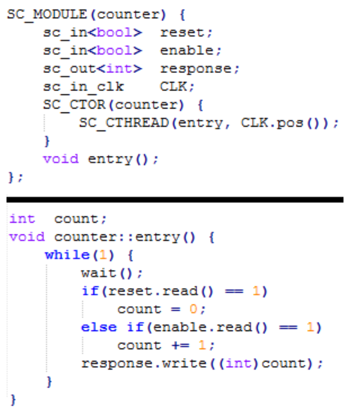


Fig. 4. Decoder class modeled in UML

The details of such methods are done through the representation of UML Activity and State diagrams, for each system functionality: Activity diagrams describe its basic and alternative flows; whereas State diagrams model the behavior of classes from the concept of inheritance, nestling and concurrency.

Once the Class diagram is finished, the Papyrus platform helps us to automatically generate C++ code, remaining only to the programmer the task of manually adjusting her/his code with some small details (e.g. setting local variables, inserting C++/SystemC directives as *#include<systemc.h>* into .cpp files, etc.) Fig. 5 presents an example of this mapping when taking into account the Timer/Counter peripheral of the 8051 microcontroller.



```
SC_MODULE(counter) {
    sc_in<bool>  reset;
    sc_in<bool>  enable;
    sc_out<int>  response;
    sc_in_clk    CLK;
    SC_CTOR(counter) {
        SC_CTHREAD(entry, CLK.pos());
    }
    void entry();
};
```

```
int  count;
void counter::entry() {
    while(1) {
        wait();
        if(reset.read() == 1)
            count = 0;
        else if(enable.read() == 1)
            count += 1;
        response.write((int)count);
    }
}
```

Fig. 5. Simplified integers counter in SystemC (.h (top)/.cpp (bottom) codes)

VI. Conclusions and Future Works

Our contribution with this work is the presentation of a framework able to help students to construct embedded systems from combining learning approaches (with PBL) and advanced software engineering concepts (with visual modeling, object-oriented languages, methodologies and processes).

Considering the pedagogical scaffolding, the introduction of PBL in this undergraduate research project produced effective changes in the learning process of the participating students. Firstly, the insertion of each student in an active role in the construction of her/his own knowledge (it was noticed when students feel confident they participate and become more interested [18]). Secondly, students having the freedom to choose different strategies to solve their problems may become more engaged in the learning process, since they are encouraged to explore topics of interest, acquiring self-evaluation, communication, collaboration and problem-solving skills. Thirdly, the constant practice of always associating a subject addressed in the research with theoretical and practical results of works developed in industry as well as in academy highly motivated the participating students.

Considering software engineering approaches, recent works point to the design of embedded systems from high-level technologies supporting modeling, model transformation, verification and validation features. Among them, the use of UML to specify hardware design from requirements to source code generation is still representing a trend nowadays. In particular, an interesting paper in this research domain quantifies such observation through some recent resulting-oriented papers published from 2008 to 2014 in reputable scientific databases as ACM, IEEE, Elsevier and Springer [19]. Following this trend, our case study was developed from its UML modeling to its SystemC implementation through the use of several software engineering tools, methodologies and processes.

Although the presence of experienced students in a project scaffolded by PBL and using advanced software engineering concepts is needed, the participating students showed a positive consensus about the framework applied in this project thanks to, according to their opinion, some learning mechanisms reinforced in this research (e.g., debate of ideas, change of roles, design of artifacts, interpretation of findings and drawing of conclusions).

In the same way we could observe the dedication of the students participating in this research when applying PBL as a learning process, we could also remark their commitment when using advanced software development approaches to design hardware projects, mainly because such students realized they were able to solve problems in the computer engineering domain from the use of techniques originally belonging to the software engineering field.

However, it is also necessary to indicate the issues faced by these students: they recognize some difficulties to initialize the project and of dealing with time constraints ("only 12 months of project duration?!"). The students' feedback was gathered from "think aloud" observations and meetings conducted weekly, where the project results were presented, its schedule evaluated, and the next activities to be implemented, deliberated.

As future work we intend to apply this framework in UESC Computer Science advanced courses as a way to motivate more students of this educational background to design complex embedded systems. Also, we intend, as a new undergraduate research project, integrating the case study developed in this work as a core in a Network-on-Chip (NoC) responsible to parallelize the compression of colored images.

ACKNOWLEDGMENTS

The authors with to thank the Fundação de Amparo à Pesquisa do Estado da Bahia (FAPESB) for encouraging this research within the Universidade Estadual de Santa Cruz; and also with to thank the ipPROCESS team for taking time to clarify our doubts during the deployment of its process in our research project.

REFERENCES

[1] M. S. M. Lima, “ipPROCESS: Um processo para desenvolvimento de IP-Cores com implementação em FPGA,” M.S. thesis, Dept. Comput. Scienc., Univ. Fed. Pernambuco, Recife, 2005.

[2] P. L. Sobreira et al., “Competição como uma técnica motivacional no ensino de Arquitetura de Computadores,” in *Workshop sobre Educação em Arquitetura de Computadores, (held in conjunction with the) 19th IEEE Int. Symp. Computer Architecture and High Performance Computing*, Gramado, 2007, vol.1, pp. 39-42.

[3] M. A. Mazidi, J. G. Mazidi and R. D. McKinlay, “The 8051 microcontroller and embedded systems using Assembly and C”. 2nd ed., Pearson, 2007.

[4] M. Rashid, "System Level Approach for Computer Engineering Education," Int. J. Eng. Educ., ser. 1(A), vol. 31, pp. 141-153, 2015.

[5] J. E. Christensen and K. Ribu, “Integration of students in the teaching process,” in *Proc. 9th Int. Conf. Engineering Education*, Mayagüez, 2006, pp. 21-26.

[6] Q. Zhu et al., “An object-oriented design process for system-on-chip using UML,” in *Proc. 15th Int. Symp. System Synthesis*, Kyoto, 2002, pp. 249-254.

[7] A. Habibi and S. Tahar, “A survey on system-on-a-chip design languages,” in *Proc. 3rd IEEE Int. Workshop System-on-Chip for Real-Time Applications*, Calgary, 2003, pp. 212-215.

[8] A. Tsikhanovich et al., “Object-oriented techniques in hardware modeling using SystemC,” in *Proc. 1st Annu. Northeast Workshop Circuits and Systems*, Montreal, 2003, pp. 97-100.

[9] G. Martin et al., “Embedded UML: a merger of real-time UML and co-design,” in *Proc. 9th Int. Symp. Hardware/Software Codesign*, Copenhagen, 2001, pp. 23-28.

[10] Q. Zhu et al., “Integrating UML into SoC Design Process,” in *Proc. Design, Automation and Test in Europe*, Munich, 2005, vol. 2, pp. 836-837.

[11] F. Boutekkouk, “Automatic SystemC code generation from UML models at early stages of systems on chip design,” Int. J. Comput. Appl., vol. 8(6), pp. 10-17, Oct. 2010.

[12] J. A. S. Villa and D. Serna, “SystemC code generation from UML for wireless sensor networks design,” in *Proc. Int. Conf. Modeling, Simulation and Visualization Methods*, Las Vegas, 2011, pp. 53-60.

[13] F. Santos et al., “ipPROCESS: A usage of an IP-core development process to achieve time-to-market and quality assurance in a multi project environment,” in *Proc. IP-based System Design Conference*, Grenoble, 2008.

[14] H. Hallal et al., “Experiments in modeling integrated circuit blocks by UML,” in *Proc. Int. Workshop IP-based Synthesis and SoC Design*, Grenoble, 1999.

[15] D. Jain. (2004). *Analysis and VHDL modeling of 8051-microcontroller using determinant-functional object modeling (D-FOM) approach* [Online]. https://www.codeproject.com/articles/7757/analysis-and-vhdl-modeling-of-microcontroller (accessed 19 December 2016).

[16] L. Kesen, "Implementation of an 8-bit microcontroller with SystemC," M.S. thesis, Grad. School Nat. App. Scienc. Mid. East Tech. Univ., Ankara, 2004.

[17] Papyrus. http://www.eclipse.org/papyrus (accessed 19 December 2016).

[18] P. J. Teller et al., “Combining learning strategies and tools in a first course in computer architecture,” in *Workshop Computer Architecture Education, (held in conjunction with the) 30th Int. Symp. Computer Architecture*, Anchorage, 2003. doi =10.1145/1275521.1275533

[19] M. Rashid et al., “Towards the tools selection in model based system engineering for embedded systems – A systematic literature review,” J. Syst. and Soft., vol. 106, pp. 150-163, 2015.