USING GAMES IN THE TEACHING OF DIGITAL SYSTEMS AND OF COMPUTERS ARCHITECTURE

Pedro José Guerra de Araújo
IT - Institute of Telecommunications, Department of Computer Science, University of Beira Interior
Av. Marquês d'Àvila e Bolama 1, 6200 Covilhã, Portugal

João Vieira Baptista
IESIG – Instituto de Estudos Superiores Isidoro da Graça, São Vicente, Cabo Verde

Keywords: Computer science, learning methodologies, computer games.

Abstract: This work presents a pedagogic experience regarding the use of games as support examples on Digital Systems and Computers Architecture teaching. For this work it is not the game itself that is important, but the technologies involved in the construction of the game. Examples of each one of these technologies are presented here through the construction of three different versions of the same game. The first version is totally made in hardware using digital integrated circuits, the following is accomplished by programming using the Assembly language and finally a third version using a high-level language. The importance of these examples lies in the fact that each one of these versions allows one to exemplify techniques that can also be found in other types of computer programs. Besides capturing the attention and the students' interest immediately, the games are good examples to serve as support to the concepts' exposition done by the teacher in the classroom.

1 INTRODUCTION

Just as it happens in other scientific areas, the computer science teaching faces great challenges. Nowadays, the educational system has to compete with simplified forms of obtaining knowledge such as magazines, television and the Internet.

The diffusion of these media and the easy access to several equipments (PC's, pocket calculators, game machines, mobile phones) frequently creates the illusion in the students that it is not necessary to dominate the theoretical knowledge that it is in the base of the operation of those technologies. Seemingly, what is necessary is to know how to use, instead of knowing how to do. Another problem is that, due to time constraints, the examples used in the classroom are just isolated fragments of larger programs, or they are simplified versions of more complex digital circuits, unable to attract the attention and the students' interest. In a simpler way, using the Internet, the students can easily obtain complete systems at zero cost. This creates great challenges to the teacher on how to convince the students about the importance of the matters that they should learn.

The games are good examples to help to change this situation. Indeed, in terms of computers technology, games present important challenges. They force the use of sophisticated programming techniques and the interface with the hardware, often leading to the development of new devices for data acquisition and information visualization, as well as new and better processing algorithms and data structures, more efficient in terms of time and space. This means that the development of games involves the deep knowledge of the physical structure of the machines and of related programming techniques. Besides the technical knowledge, games appeal to the imagination mainly in the phase of objectives and rules definition, being an activity which is very appreciated by the students.

This way, the games are excellent examples that help the teacher to expose the subjects to the students, in a "learning-by-doing" process.

José Guerra de Araújo P. and Vieira Baptista J. (2007).
USING GAMES IN THE TEACHING OF DIGITAL SYSTEMS AND OF COMPUTERS ARCHITECTURE.
In *Proceedings of the Ninth International Conference on Enterprise Information Systems - HCI*, pages 5-10
DOI: 10.5220/0002379900050010
Copyright © SciTePress

2 TEACHING METHODOLOGIES

Bormida et al., 1997, pointed four applied methodologies on the digital logic and computers architecture teaching: expositive, demonstrative, interactive and practical.

In the expositive methodology, the presentation of theory, concepts and other information is based on multimedia hypertext making use of animations. In the demonstrative methodology, the concepts previously introduced are reinforced through examples of practical implementations. A simulator can be used, which can be controlled by the student by setting the values of some of the input variables and to observe the behaviour of the outputs. In the interactive methodology the student make choices, describe or calculate something. They therefore imply an evaluation of the student's answers. Finally, in the practical methodology, the student should accomplish a project to acquire design capabilities. Conceiving circuits and systems is generally more difficult than understanding and analyzing them and requires different and complementary skills.

In the expositive and demonstrative approaches, the student might go through the subjects without paying the necessary level of attention to assimilate the topics presented. Instead, in the case of interactive and practical methodologies, they are forced to make decisions, leading to a larger involvement in the study. The use of tests and exercises with the aid of simulators is a good methodology to evaluate if the knowledge was really learned. In agreement with those authors, the use of simpler tools is the most appropriate. As the conclusion of their study shows, the students generally prefer the experimental practice instead of reading hypertextual material or watching simple animation.

The development of games is an excellent process to put into practice each of these methodologies. As a matter of fact it is possible to use games for to exemplify most of the techniques involved in the development of many other types of computer programs. Besides the hardware, these techniques include the study and the control of input/output devices, of the interaction between the software and the hardware, of the organization of the central processing units, of the memory management, of the algorithms and data structures, of the graphics programming, etc.

In the next chapters a small game is used to explain these ideas.

3 THE SUM/SUBTRACTION GAME (SSG)

This game has two main objectives, the first is the construction of the game itself and the second is its use. From the construction's point of view, the game exemplifies each one of the methodologies presented in the previous paragraphs. It helps the teacher in the exhibition of the concepts concerning digital logic and simultaneously allows the concepts' demonstration. Afterwards the students are invited to accomplish the game in a simulator to test its operation and to correct some mistakes. Next, they should implement the circuit using several technologies.

From the point of view of its use, the game intends to be an auxiliary to the students' training in binary arithmetic. Namely, it allows the practice of conversions from binary to decimal and from decimal to binary, but also the training of the arithmetic operations sum and subtraction using the two's complement notation. To win the game, the player must be able to quickly do these calculations and conversions.

Figure 1 show the components of the game, which rules are:

- After the "Start" button is pressed, the circuit generates an hexadecimal random value in the range from - Fh (-15) to Fh(15), that is shown in the "Number" display;
- Since that moment, the player has a short interval of time to introduce in "Binary Value" input the value in binary code corresponding to the symmetric of the "Number" value (including the sign bit); the length of time to introduce the answer is controlled by the "Speed" selector;
- After pressing the "Test" button the circuit adds the values in "Number" and in "Binary Value" ("Number" + "Binary Value"), and then:
  - if the value of the sum is zero (correct answer) the circuit marks a point in "Ok" in favour of the player and also increases the total number of attempts in "Total";
  - if the value of the sum is not zero (wrong answer), or if the time to answer was elapsed without player's answer, only the total number of attempts in "Total" is increased;
- The difference between the "Total" value and the "Ok" value indicates the number of wrong answers.

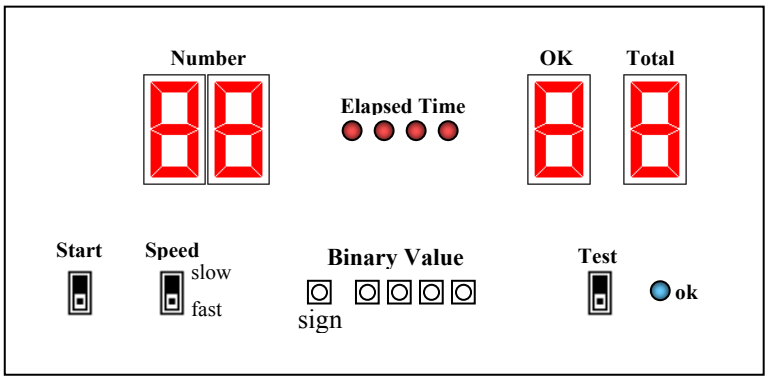

Figure 1: Components of SSG (Sum/Subtraction Game).

From now on the construction of the game will allow one to exemplify almost all of the concepts related with the operation of a personal computer, at hardware and software levels. It also serves to show the implementation of a same idea using different technologies. For that, the game will be initially implemented in hardware, using integrated circuits, followed by an implementation in machine language and finally, using a high-level language. In the next paragraphs the contributions of these different implementations of the game on computer technologies' teaching will be discussed.

3.1 Learning Digital Systems

One of the most common techniques used on the teaching of digital systems is the use of simulators, that there are several examples (Ferreira et al., 2003). They allow the modelling of digital circuits as logic gates, counters, registers and memory, and also reproduce real systems such as elevators, traffic lights or the doors and windows of a house. By using them it is possible to develop small and medium size projects such as elevators controllers, traffic lights management systems or residential alarm systems.

The example of the game here presented was built using the simulator Digital Works (WWW1, 2006). Although being a quite simple simulator, it allows modelling most of the concepts presented in an introductory course to digital systems. The schematic diagram of the circuit is presented in figure 2, while figure 3 shows the picture of the correspondent hardware implementation.

By analyzing the circuit it can be seen that although the game's objective is very simple, it uses most of the digital components found in more complex systems. The components used by the game are, among others, logic gates, counters, registers, memory, input/output devices (switches, leds and seven-segments displays), decoders, multiplexes, arithmetic circuits (sum/subtraction), comparators, clock generator and frequency dividers. The interest of the game comes from the fact that with only one circuit the operation of all these components can be explained.

3.2 Learning Computer Architecture

Another important subject is the teaching of the Assembly language. For that, several simulators exist (Ferreira et al., 2003). Once acquired the essential concepts on digital circuits, the students should understand how these circuits are interconnected to form larger blocks, such as central processing units (CPU), memory controllers, communication channels (buses), input/output ports and finally a complete personal computer (PC). They should also understand that those blocks dialogue between them using a low-level language formed by groups of binary digits, called machine language.

The knowledge and the practice of this language allow the students to understand the data flows that circulate inside the machine as well as the way they are processed. By this way, the students consolidate their knowledge about circuits' operation which allows them to develop techniques for programs' optimization.

The programming of the SSG game in Assembly language was made using the NASM Assembler (WWW2, 2006). The program makes use of concepts such as dialogue with the operating system and the input/output system (calls to the OS and of the BIOS), routines for generation of random numbers, low-level manipulation of the keyboard, of the mouse and of the video, graphics programming, etc. Figure 4 presents the screen of the game.

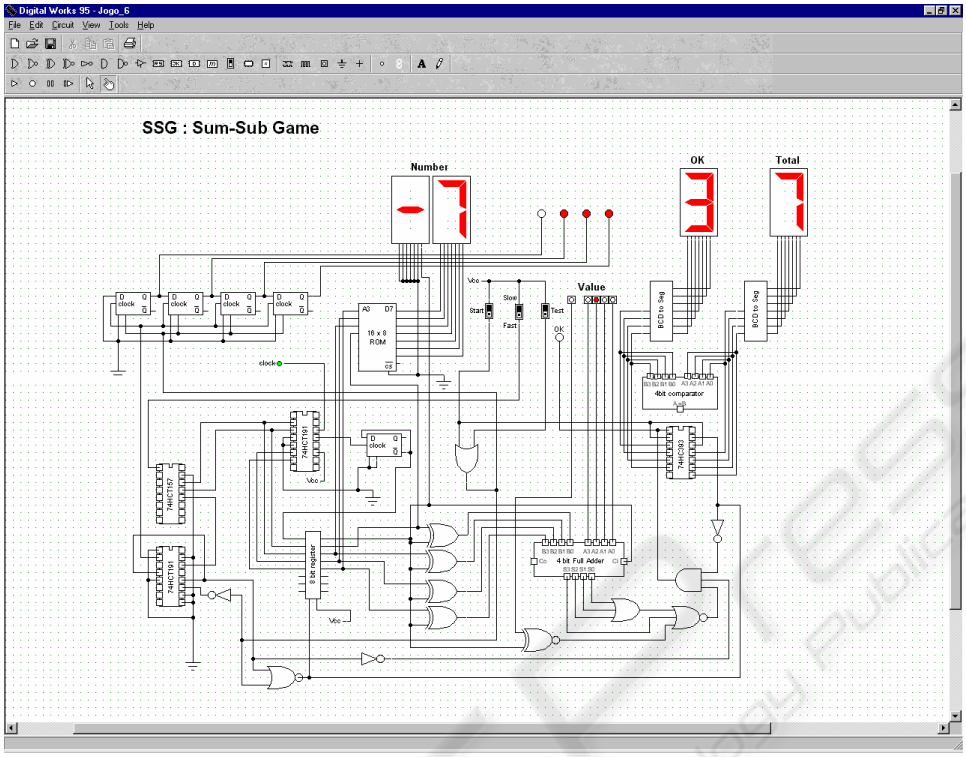

Figure 2: Schematic Diagram of SSG.

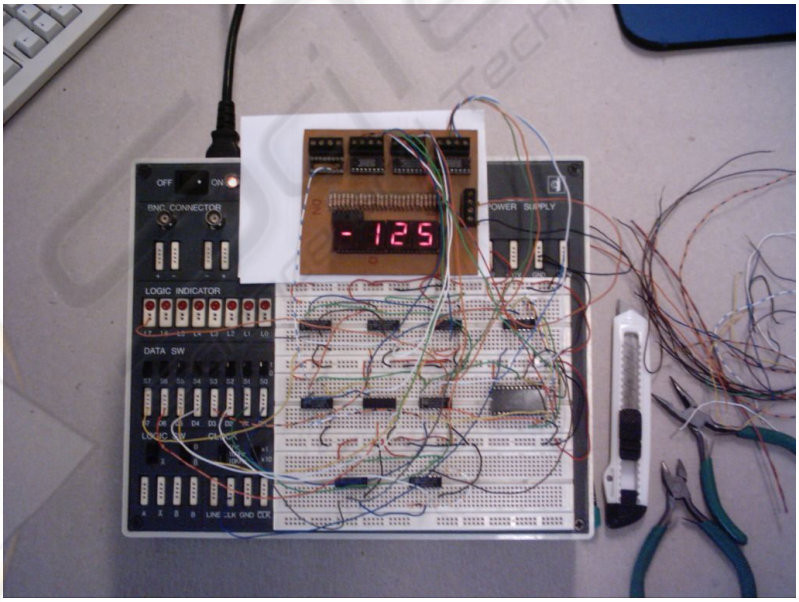

Figure 3: The hardware of SSG.

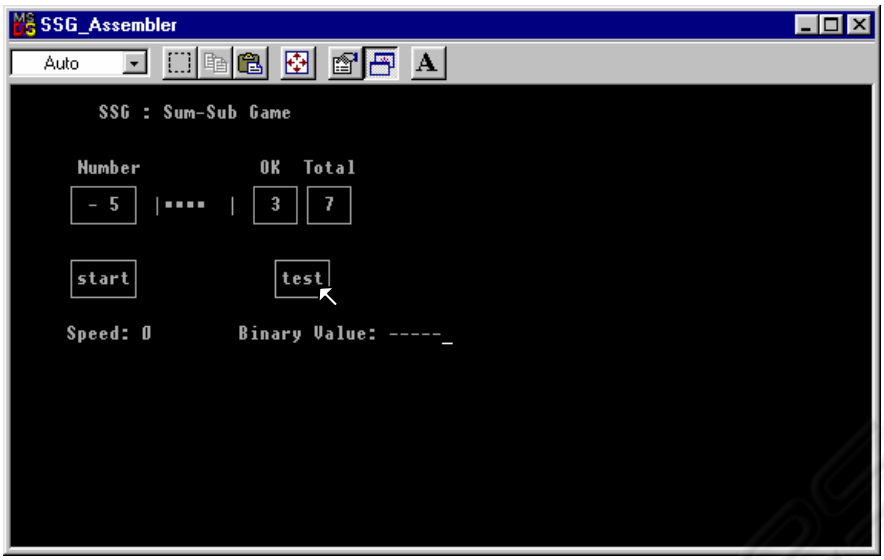

Figure 4: SSG done in Assembly (NASM).

3.3 Learning High-level Languages

Most of the first year students of an engineering course are already familiarized with some high-level programming language, like C or Pascal. But for many of them the relationship between the instructions of these languages and its execution by the hardware is unknown, even mysterious.

After the two previous versions of the game have been accomplished (in hardware and in machine language), the students have a better idea about that relationship. Namely, they already understand how the software controls the hardware.

The next step is to understand that the high-level languages are closer to the human languages and at that they provide a set of enlarged capacities including better control and data structures, much more efficient instructions and improved error handling. Besides, they hide from the programmer the complex details of the operating systems as well as the details of the hardware.

The programming of the SSG game in a high-level language was made in Object Pascal using the integrated development environment (IDE) Delphi from Borland/Inprise (WWW3, 2006). The program allows exemplifying the drawing of user interfaces, through the use of visual components as buttons and dialogue boxes, but it also exemplifies the use of special components as timers. Figure 5 presents the screen of the game.

4 CONCLUSIONS

Games constitute an excellent way to teach hardware and software technologies. The construction of a game involves the use of several techniques, such as the control of hardware for data acquisition, as well as the use of several data processing algorithms and of information visualization.

They allow the capture of the students' interest on the concepts required in its accomplishment. Firstly, the development of the original idea and the definition of the rules of the game. Secondly, the study of models and algorithms to be used (could include mathematics and physics knowledge up to now without apparent usefulness). Thirdly, the development phase of hardware and/or software and finally the user interface.

The creation of games is a task that motivates the students, easing the understanding of the concepts transmitted by the teacher in the classroom. They stimulate the students' imagination, making them propose new improvements and new functionalities. The games are also incentives in professional terms, because the students know that they constitute important business opportunities.

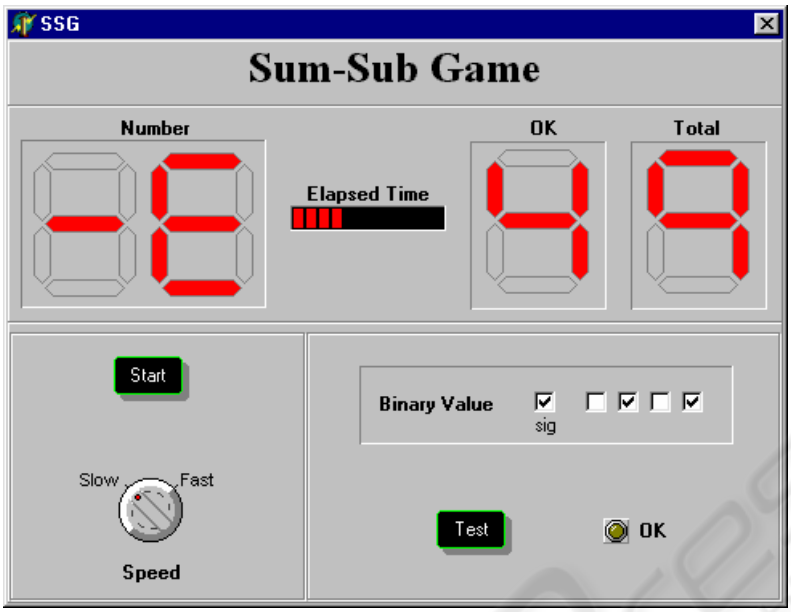

Figure 5: SSG done in Borland DELPHI.

REFERENCES

Bormida, G. D., Ponta, D., and Donzellini, G., 1997. Methodologies and tools for learning digital electronics. IEEE Transaction in Education, 40(4).

Ferreira R., Chieppe U., Freitas G., Biancardi C., 2003. Software Livre no Ensino de Sistemas Digitais e Arquitectura de Computadores, Universidade Federal de Viçosa, Departamento de Ciência da Computação.

WWW1 (2006). Digital Works, http://www.spsu.edu/cs/faculty/bbrown/circuits/

WWW2 (2006). NASM, The Netwide Assembler, http://sourceforge.net/projects/nasm/

WWW3 (2006). Delphi IDE, http://www.borland.com/us/products/delphi/index.html