The Use of ARM-Assembly Language and a Raspberry Pi 1 B+ as a Server to Improve Computer Architecture Skills

Vitor Manuel Ferreira
Instituto Politécnico de Viana do Castelo, Portugal
ferreira@estg.ipvc.pt

Pedro Pinto
Instituto Politécnico de Viana do Castelo, Portugal
INESC TEC, Porto, Portugal
pedropinto@estg.ipvc.pt

Sara Paiva
Instituto Politécnico de Viana do Castelo, Portugal
sara.paiva@estg.ipvc.pt

Maria José Azevedo Brito
Instituto Politécnico de Viana do Castelo, Portugal
Centro de Linguística da Universidade Nova de Lisboa (CLUNL), Portugal
mjazevedo@estg.ipvc.pt

Abstract

Prompting students' interest and engagement in learning environments is crucial to achieve the best results. Academia and educators in general are constantly adapting materials and methodologies in order to maximise the acquisition of contents by their students. In this case-study, a new teaching/learning methodology is presented and evaluated through a final questionnaire survey. This case-study aims to understand students' efficiency and motivation levels regarding a new teaching/learning methodology adopted in the second module of a Computer Systems and Architectures course attended by first-year Computer Sciences undergraduates. The new teaching/learning methodology relies on a specific programming language - ARMv6 assembly - to improve students' efficiency levels, and an innovative always-visible in-class mobile test scenario, implemented through a low-cost computing platform - Raspberry Pi 1 B+ - as a server, mimicking as much as possible a real-life environment, so that students believe they are working on real hardware, thus enhancing their motivation levels. The results of the questionnaire survey allowed to infer that the use of a specific programming language, such as ARMv6 assembly, coupled with a new always-visible in-class mobile test scenario were in fact efficient in raising the levels of motivation among Computer Sciences students and, consequently, improved their skills in Computer Architecture.

2012 ACM Subject Classification Computer systems organization

Keywords and phrases ARM-assembly language, Raspberry Pi, always-visible in-class mobile test scenario, Computer Architecture skills, students' efficiency and motivation levels evaluation

Digital Object Identifier 10.4230/OASIcs.ICPEC.2020.8

1 Introduction

According to Dunne (2017) [4], the best way to understand how a Computer works – more specifically, how its Central Processing Unit (CPU) works – is through the use of its assembly language, because it is "...the computer programming language closest to (its) CPU's machine language". Although an assembly language is "...unique to a particular CPU design" and, therefore, "not portable from one CPU manufacturer or model to another", what is interesting to highlight is the author's claim that:

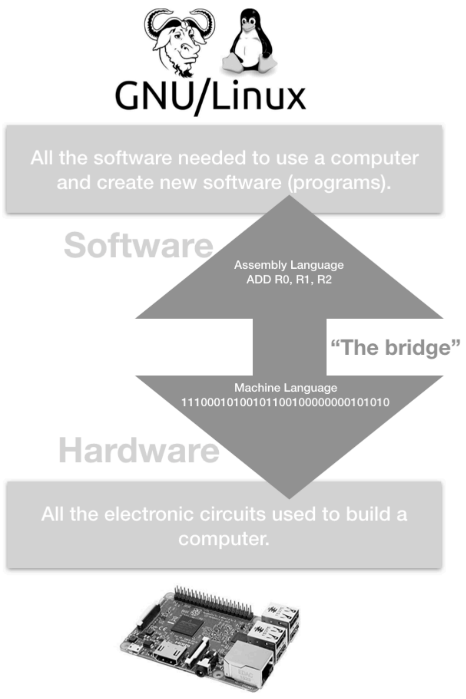

GNU/Linux
All the software needed to use a computer and create new software (programs).
Software
Assembly Language
ADD R0, R1, R2
"The bridge"
Machine Language
11100010100101100100000000101010
Hardware
All the electronic circuits used to build a computer.

Figure 1 Assembly language, the bridge between hardware and software (adapted from Dunne (2017) [4]).

> "...assembly language is primarily a bridge of understanding between programmers and computer engineers" (Figure 1).

Consequently, since the main goal of the *Computer Systems and Architectures* (CSA) course is to develop skills associated with "...the relation between software and hardware and how programming tasks are executed by hardware" [12], students will be able to learn "...computer architecture and internal processor organization through the writing of assembly programs" [1].

2 Reasons for choosing a Raspberry Pi computing platform

The choice for a low-cost Raspberry Pi computing platform, with a RISC architecture - ARM processor, was made based on Clements's (2010) [3] and Dunne's (2017) [4] claim. In fact, we could have used an emulator [7] instead of real hardware, but "...students do not want to use hypothetical hardware, because they feel it is unrealistic and does not give a true picture of the real world they will soon be entering" [3]. On the other hand, the same authors also argue that "...ARM architecture is an excellent vehicle for teaching computer architecture" [3] because it is "...one of the most popular CPUs currently in production" [4] and used worldwide; the truth is that we all carry a smartphone with a RISC processor in our pocket [8]. Moreover, according to Dunne (2017) [4], choosing a Raspberry Pi computing platform brings further advantages, such as:

1. "Professional quality" – Using a RISC ARM CPU computing platform with a GNU/Linux operating system is "...a common work environment for developing real-time embedded systems";

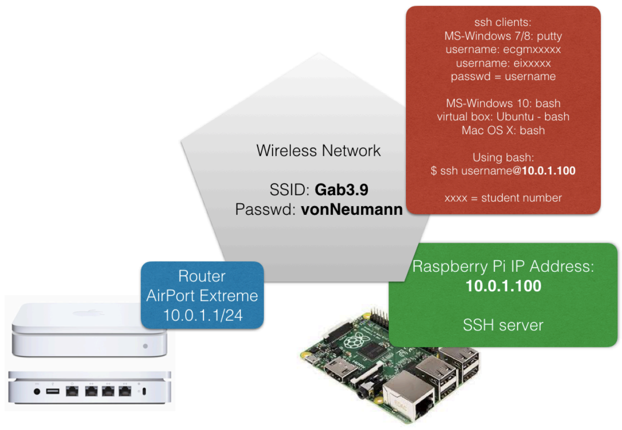


Figure 2 The new in-class scenario.

2. "Edit, Compile, Link and Execute" – Since students typically use Integrated Development Environments (IDE) to develop software, what apparently happens is that they do not realise the key steps involved in software development, namely *Edit*, *Compile*, *Link* and *Execute*. Therefore, in the CSA course, students "...explicitly perform each of these steps separately so (they) can learn the role of each program - editor, assembler, linker" [10] used to create the executable program;
3. "Inexpensiveness" – Most importantly, the low cost of such a platform will allow students to easily acquire it and be able to work outside the classroom environment.

3 The new mobile test scenario

Figure 2 shows the new mobile test scenario used in the classroom environment.

Within this test scenario, a Raspberry Pi 1 B + was permanently used in overclocking at 900 MHz, in order to be as fast as possible without compromising the system stability (Figure 3). To make this test scenario as similar as possible to a real-life environment, a Raspberry Pi was used as ssh server to be able to serve, simultaneously, about 40 students per classroom. With the purpose of keeping the test scenario in students' minds, the portable/mobile test scenario was consistently taken to the classroom instead of having a Raspberry Pi somewhere in the school's wireless network.

However, to manage to serve a practical class of approximately 40 students, it was necessary to use an adequate router. We started with a Linksys WRT54GS Wireless-G broadband router, yet, it was not able to guarantee a persistent connection with the Raspberry Pi, due to the high number of students in class. Therefore, we decided to try out another router – Apple's AirPort Extreme – and, from then on, the test scenario worked without incident.

On the server side, we created an account per student, so that each student could work on the Raspberry Pi using a ssh client at the following command line:

```
$ ssh studentUsername@10.0.1.100
```

With the entire test scenario fully operational, the teaching/learning strategy was grounded on Tanenbaum & Austin's (2013) [11, p. 1] premise:

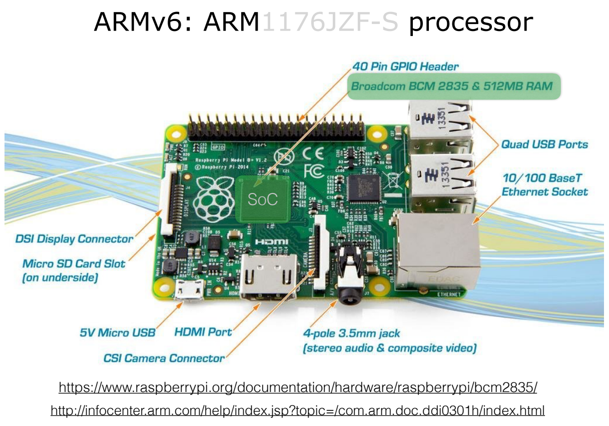

ARMv6: ARM1176JZF-S processor
40 Pin GPIO Header
Broadcom BCM 2835 & 512MB RAM
Quad USB Ports
10/100 BaseT Ethernet Socket
SoC
DSI Display Connector
Micro SD Card Slot (on underside)
5V Micro USB
HDMI Port
CSI Camera Connector
4-pole 3.5mm jack (stereo audio & composite video)
https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2835/
http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0301h/index.html

Figure 3 The Rasberry Pi 1 B+.

> "The electronic circuits of each computer can recognize and directly execute a limited set of simple instructions into which all its programs must be converted before they can be executed. These basic instructions are rarely much more complicated than (1) Add two numbers; (2) Copy a piece of data from one part of the computer's memory to another; (3) Check a number to see if it is zero."

Based on this premise, students were challenged to create three simple assembly programs capable of (1) adding two numbers, (2) transferring data from RAM to registers and registers to RAM, and (3) determining whether a number is equal to zero, respectively. At this stage, students were able to learn how to edit the source code of each program, using the vi(m) text editor; how to compile, using the GNU Assembler; how to create the executable program, using the GNU Linker; and how to see the output of each program, using the bash "Exit-Status" variable. All the commands used in each cycle are shown in Figure 4.

Noticeably, the aim of the CSA course is not to make students experts in assembly programming, but rather to allow them to learn computer architecture by writing small programs in assembly, as also claimed by Ibanez (2013) [5]:

> "The idea is not to become a(n Assembly programmer) master but understand some of the details of what happens underneath."

Therefore, students were instructed to follow the first five online tutorials from the above author [5, 6], covering Tanenbaum & Austin's (2013) [11, p. 1] premise.

Finally, to actually see each of these basic instructions in action, all the assembly programs were run, step-by-step, through the GNU Debugger [9].

4 Perception of students' efficiency and motivation levels

In order to perceive and understand students' efficiency and motivation levels regarding the adopted teaching/learning methodology, a questionnaire survey was used to collect data related to:

1. Level of effort;
2. Contribution to learning process;
3. Instructor's skill and responsiveness;
4. Course content.

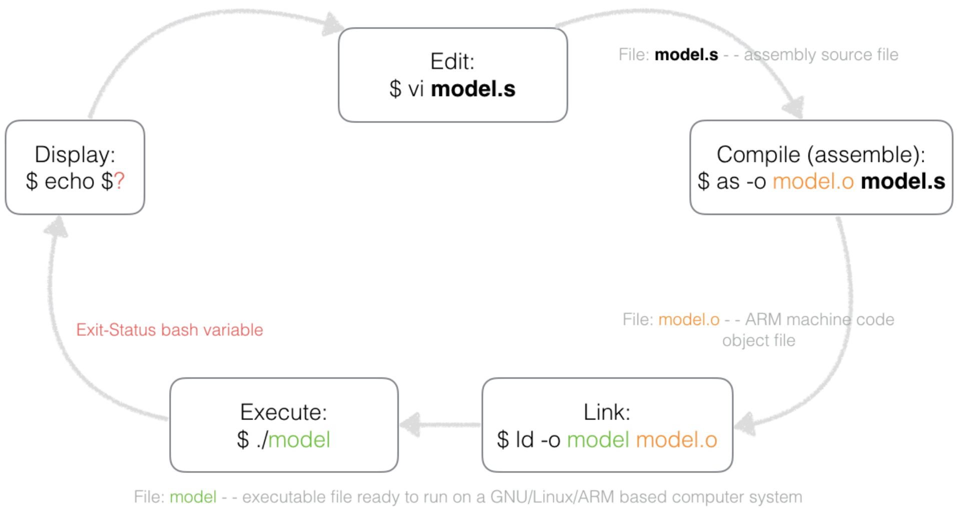

Edit:
$ vi model.s
File: model.s - - assembly source file
Compile (assemble):
$ as -o model.o model.s
File: model.o - - ARM machine code object file
Link:
$ ld -o model model.o
Execute:
$ ./model
File: model - - executable file ready to run on a GNU/Linux/ARM based computer system
Exit-Status bash variable
Display:
$ echo $?

Figure 4 Workflow to test each assembly program into the bash GNU/Linux Operating System.

Since the first two items measure Satisfaction, we used the following scale ratings: Poor, Fair, Satisfactory, Very good and Excellent. As for the last two items, which measure Attitudes and Opinions, our choice was the Likert scale [2]: Strongly disagree, Disagree, Neutral, Agree or Strongly agree.

At the end of the questionnaire survey, two simple open-ended questions were added. The first question focused on the most useful or valuable aspects of the course module, whereas the second one elicited suggestions on how to improve the course module.

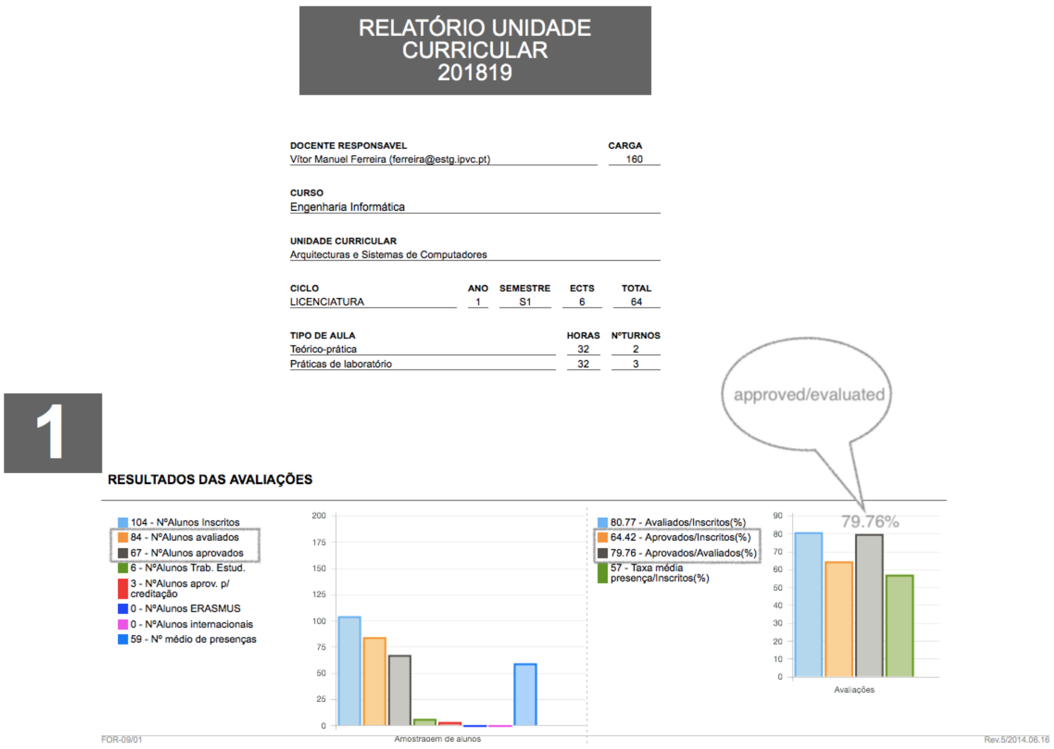
Furthermore, to find out how many students actually bought a Raspberry Pi to work outside the classroom environment, a final Yes/No question was asked.

In addition to the questionnaire survey, for purposes of comparison and attestation of the results obtained as regards students' efficiency levels, we also took into consideration the Academic Registry data included in the official CSA course final reports. These reports reflect a final assessment (FA) based on both practical and theoretical approaches. On the practical side (P), students had to submit a technical-scientific report for each practical assignment; on the theoretical side (T), students had to take a written test. The formula for this final assessment is as follows: $FA = 0.5 * P + 0.5 * T$.

5 Sample selection

The students who participated in this study were attending two Computer Sciences undergraduate degrees at the Polytechnic Institute of Viana do Castelo, namely Computer Engineering (CE) and Computer Graphics and Multimedia Engineering (CGME) (Figures 5 and 6).

In the academic year of 2018/2019, the total number of students from these two undergraduate degrees evaluated in the CSA course was precisely 100 (67 CE students and 33 CGME students). However, as a previously established inclusion criterion, students would



RELATÓRIO UNIDADE CURRICULAR 201819

DOCENTE RESPONSAVEL	CARGA
Vítor Manuel Ferreira (ferreira@estg.ipvc.pt)	160

CURSO
Engenharia Informática

UNIDADE CURRICULAR
Arquitecturas e Sistemas de Computadores

CICLO	ANO	SEMESTRE	ECTS	TOTAL
LICENCIATURA	1	S1	6	64

TIPO DE AULA	HORAS	NºTURNOS
Teórico-prática	32	2
Práticas de laboratório	32	3

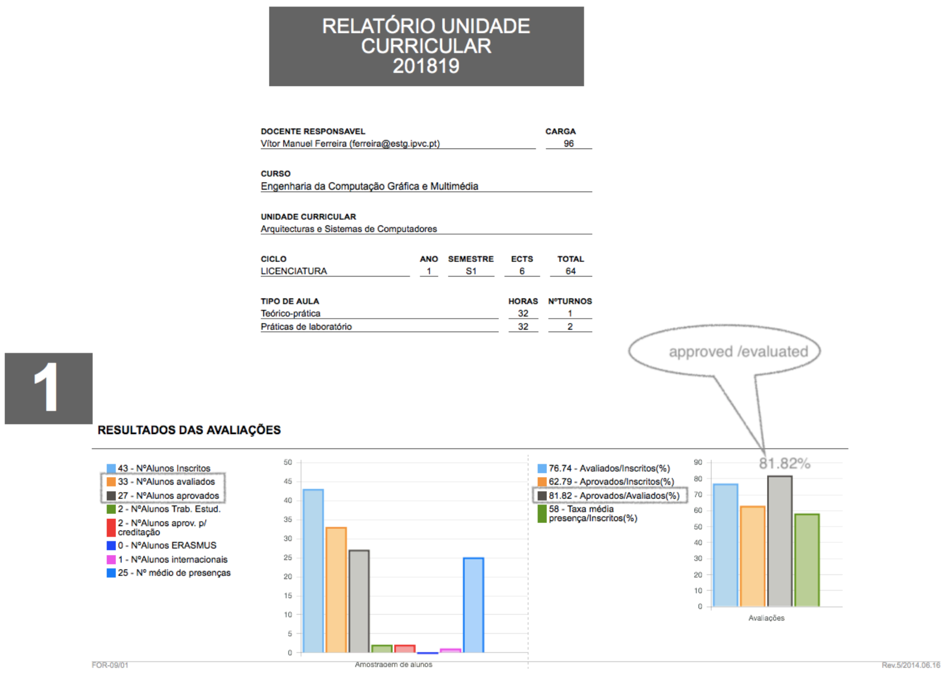
1

RESULTADOS DAS AVALIAÇÕES

- 104 - NºAlunos Inscritos
- 84 - NºAlunos avaliados
- 67 - NºAlunos aprovados
- 6 - NºAlunos Trab. Estud.
- 3 - NºAlunos aprov. p/ creditação
- 0 - NºAlunos ERASMUS
- 0 - NºAlunos internacionais
- 59 - Nº médio de presenças

- 80.77 - Avaliados/Inscritos(%)
- 64.42 - Aprovados/Inscritos(%)
- 79.76 - Aprovados/Avaliados(%)
- 57 - Taxa média presença/Inscritos(%)

Figure 5 2018/2019 CSA final report concerning the Undergraduate degree in Computer Engineering (CE) – Data retrieved from the official Academic Registry (no English version available).



RELATÓRIO UNIDADE CURRICULAR 201819

DOCENTE RESPONSAVEL	CARGA
Vítor Manuel Ferreira (ferreira@estg.ipvc.pt)	96

CURSO
Engenharia da Computação Gráfica e Multimédia

UNIDADE CURRICULAR
Arquitecturas e Sistemas de Computadores

CICLO	ANO	SEMESTRE	ECTS	TOTAL
LICENCIATURA	1	S1	6	64

TIPO DE AULA	HORAS	NºTURNOS
Teórico-prática	32	1
Práticas de laboratório	32	2

1

RESULTADOS DAS AVALIAÇÕES

- 43 - NºAlunos Inscritos
- 33 - NºAlunos avaliados
- 27 - NºAlunos aprovados
- 2 - NºAlunos Trab. Estud.
- 2 - NºAlunos aprov. p/ creditação
- 0 - NºAlunos ERASMUS
- 1 - NºAlunos internacionais
- 25 - Nº médio de presenças

- 76.74 - Avaliados/Inscritos(%)
- 62.79 - Aprovados/Inscritos(%)
- 81.82 - Aprovados/Avaliados(%)
- 58 - Taxa média presença/Inscritos(%)

Figure 6 2018/2019 CSA final report concerning the Undergraduate degree in Computer Graphics and Multimedia Engineering (CGME) – Data retrieved from the official Academic Registry (no English version available).

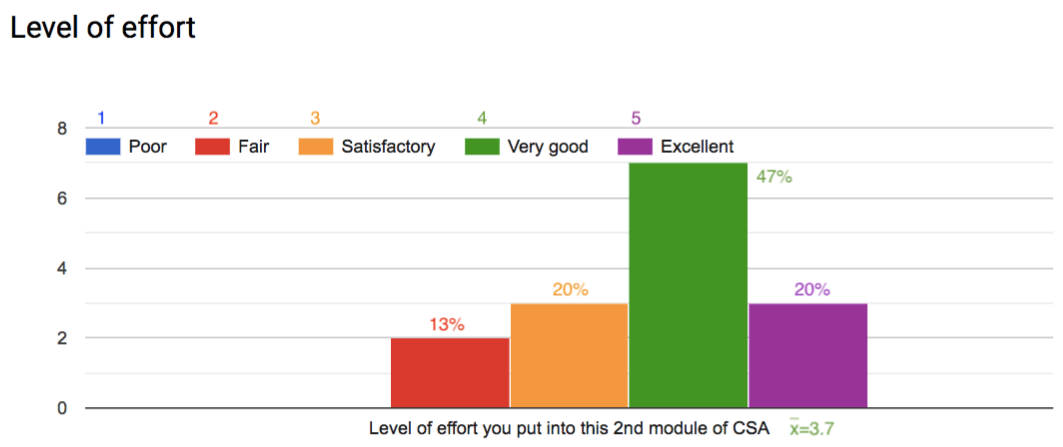


Figure 7 Self-reported level of effort invested in this 2nd module of CSA (15 responses).

need to have achieved a minimum final grade of 13 in a 0–20 scale in order to be eligible to participate in the study. Therefore, only 44 students were invited to participate, representing 44% of all the assessed students. Among these 44 participants, we obtained 15 valid answers, accounting for 15% of all the students evaluated in the course.

6 Analysis and discussion of the data collected

As both CSA final reports (Figures 5 and 6) demonstrate, overall, 80% of the evaluated students were approved. Although indirectly, this fact allows us to infer that the level of students' efficiency is in line with the students' self-reported level of effort (Figure 7) and the replies given as regards the contribution of the course to their learning process (Figure 8).

The data collected in the second item of the questionnaire survey, filled in by the 15-student sample with the main goal of retrieving evidence of the contribution of the CSA course to the students' learning process, are summarised in Figure 8 and provide evidence of: (1) *Level of skills/knowledge at start/end of the course* - although students referred that the level of knowledge at the beginning of the course was satisfying ($MEAN = 2.6$), at the end they rated it as very good ($MEAN = 4.1$); (2) *Level of skills/knowledge required to complete the course* and (3) *Contribution of the course to students' skills/knowledge* - although most students confessed that the course is demanding ($MEAN = 3.7$), most students also agreed that the final level of knowledge acquired was a result of having attended the course ($MEAN = 4.2$).

The data collected in the third item of the questionnaire survey, filled in by the 15-student sample with the main goal of revealing evidence of the instructor's skill and responsiveness, are summarised in Figure 9 and provide evidence of: (1) *Instructor's Effectiveness* - most students agreed (47%) and strongly agreed (33%) that the instructor was an effective lecturer/demonstrator; (2) *Quality of online tutorials used* - most students agreed (53%) and strongly agreed (33%) that the tutorials used were clear and well organized; (3) *Level of motivation and interest provided by the new testing scenario with Raspberry Pi* - most students agreed (53%) and strongly agreed (37%) that using a Raspberry Pi as a server, simulating a real-life environment, stimulated their interest; (4) *Useful feedback and prompt grades output* - most students agreed (53%) and strongly agreed (27%) that grading was prompt and that they had useful feedback.

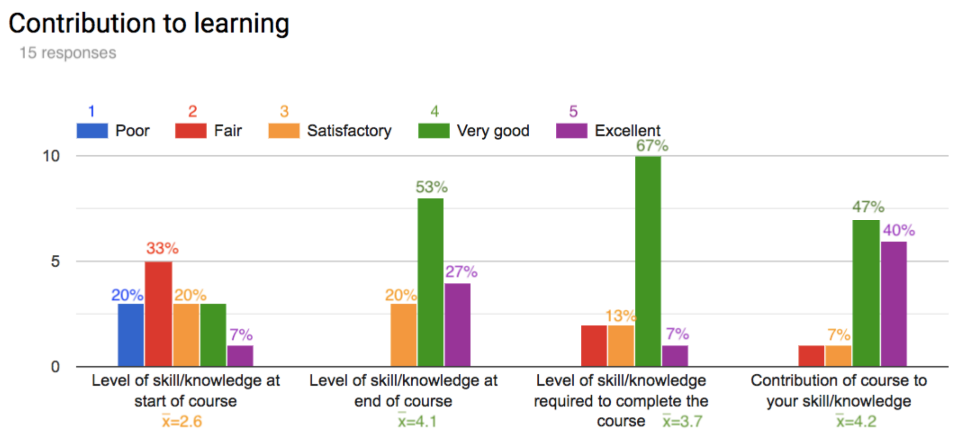


Figure 8 Contribution to the learning process (15 responses).

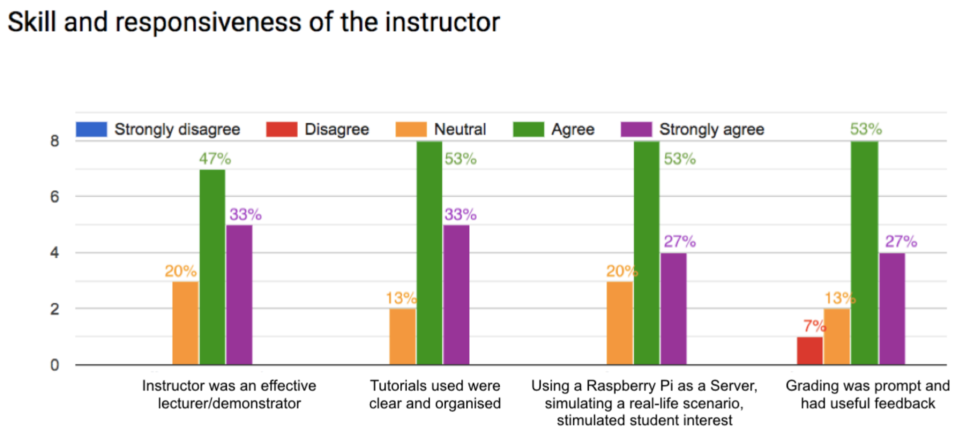


Figure 9 Instructor's skill and responsiveness (15 responses).

The data collected in the fourth item of the questionnaire survey, filled in by the 15-student sample with the main purpose of collecting evidence of Course Content, are summarised in Figure 10 and offer evidence of: (1) *Clarity of the final skills to be achieved* - most students agreed (60%) and strongly agreed (33%) that the learning skills were clear; (2) *Course content organization and planning* - most students agreed (27%) and strongly agreed (53%) that the course contents were well planned and organized; (3) *Course workload* - most students agreed (57%) and strongly agreed (20%) that the course workload was appropriate; (4) *Level of students' participation according to the course organization* - most students agreed (60%) and strongly agreed (27%) that the course was organized in a way that allowed all students to fully participate.

For the two open-ended questions (with non-mandatory answer) - (1) *What aspects of this course module were most useful or valuable?* and (2) *How would you improve this course module?* - summarised in Figure 11, we only obtained 2 answers for the first question (1) and 1 answer for the second (2): (1) "*Understanding how a computer really works, and learning a bit of low level programming*" and "*The use of tutorials for students' self-learning*" were, in the participants' opinion, the most useful and valuable aspects of this course; (2) this course module can be improved with "*More exploration of assembly programming*".

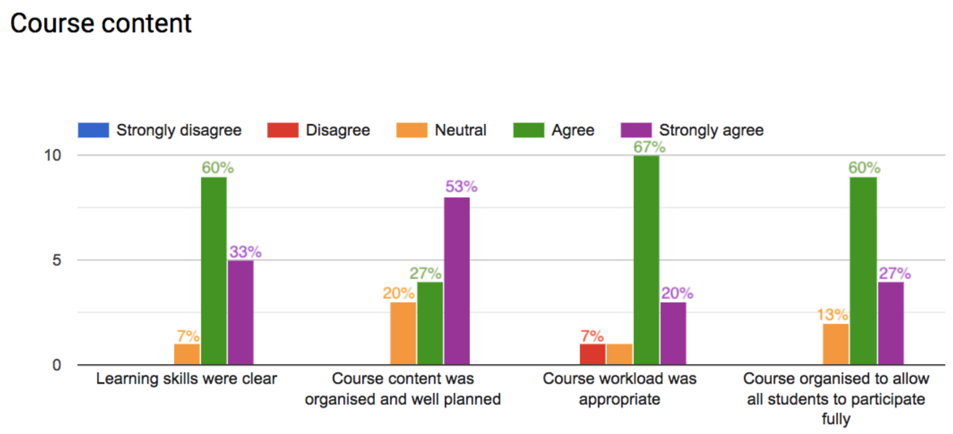
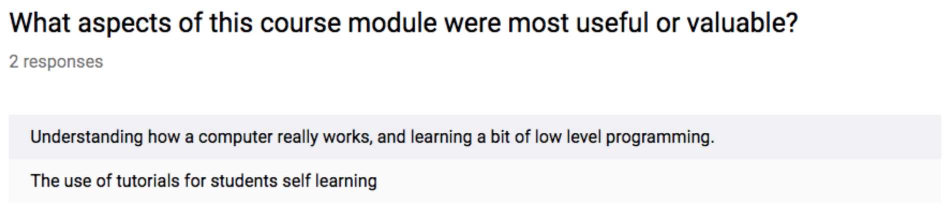


Figure 10 Course content (15 responses).

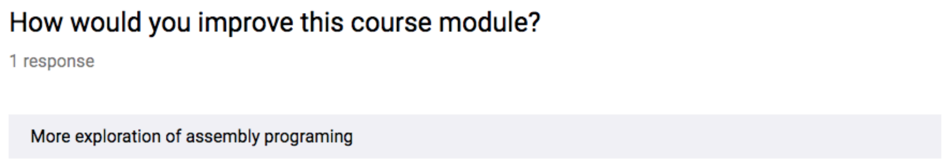


What aspects of this course module were most useful or valuable?

2 responses

Understanding how a computer really works, and learning a bit of low level programming.

The use of tutorials for students self learning



How would you improve this course module?

1 response

More exploration of assembly programing

Figure 11 Two optional open-ended questions.

Regarding the last question of the questionnaire survey, filled in by the 15-student sample with the main goal of bringing to light evidence whether the low-cost of the Raspberry Pi platform allowed them to buy it so that they could work outside the classroom environment, data are summarised in Figure 12 and demonstrate that more than half of the students (53%) bought a Raspberry Pi to work from home.

7 Conclusions

Despite the relevance of the results obtained in this study and discussed in the previous section, it is important to underline that we only obtained feedback from 34% of the 44 students invited to answer the survey. Therefore, in order to consolidate the positive results, it would be justifiable to reinforce the previous invitation to participate in the questionnaire survey, in order to obtain the answers from the remaining non-responding students and, thus, prove that this sample is indeed adequate to draw valid conclusions. In any case, we are confident and very convicted that the results obtained in this study clearly show that

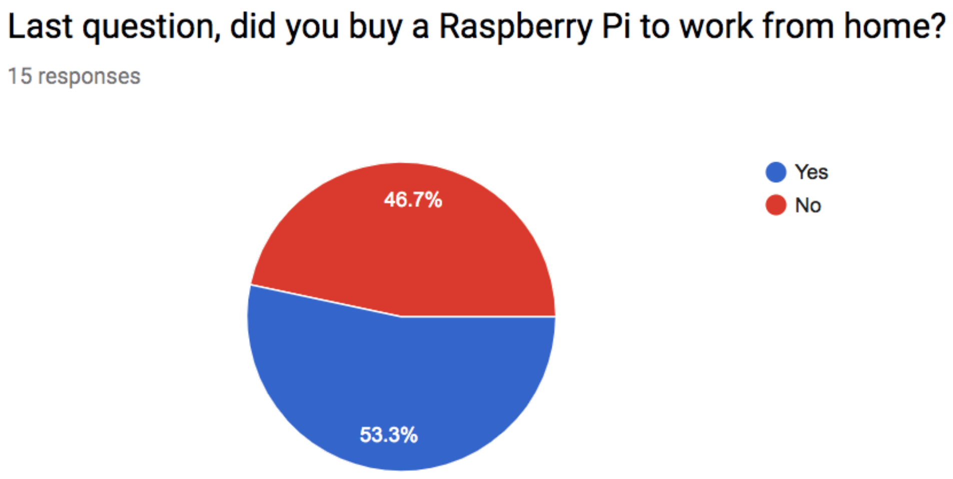


Figure 12 Last question, regarding the purchase of a Raspberry Pi to work from home (15 responses).

the teaching/learning strategy carried out in the second module of the CSA course was in fact efficient in raising the levels of motivation and interest among our Computer Sciences students, and that the use of a test scenario always present and visible in the classroom and based on a low-cost platform, such as Raspberry Pi, made all the difference. If any doubt persists, an approval rate of 80% of all the effectively evaluated students is certainly a very good indicator of the success of the teaching/learning strategy adopted in this second module of *Computer Systems and Architectures*, attended by first-year undergraduate students at the Polytechnic Institute of Viana do Castelo.

References

1 Patricio Bulić, Veselko Guštin, Damjan Šonc, and Andrej Štrancar. An FPGA-based integrated environment for computer architecture. *Computer Applications in Engineering Education*, 21(1):26–35, 2013. doi:10.1002/cae.20448.

2 Seung Youn (Yonnie) Chyung, Katherine Roberts, Ieva Swanson, and Andrea Hankinson. Evidence-based survey design: The use of a midpoint on the likert scale. *Performance Improvement*, 56(10):15–23, 2017.

3 A. Clements. ARMs for the poor: Selecting a processor for teaching computer architecture. In *2010 IEEE Frontiers in Education Conference (FIE)*, pages T3E–1–T3E–6, October 2010. doi:10.1109/FIE.2010.5673541.

4 Robert Dunne. *Assembly Language Using the Raspberry Pi: A Hardware Software Bridge*. Gaul Communications, 2017.

5 R. F. Ibáñez. ARM assembler in Raspberry Pi - Chapter 1,2,3,4 and 5, 2013. URL: https://thinkingeek.com/2013/01/09/arm-assembler-raspberry-pi-chapter-1/.

6 R. F. Ibáñez and William J. Pervin. *RASPBERRY PI ASSEMBLER*. Online, 2017. URL: http://tiny.cc/v20m7y.

7 G. Malhotra, N. Atri, and S. R. Sarangi. emuARM: A tool for teaching the ARM assembly language. In *2013 Second International Conference on E-Learning and E-Technologies in Education (ICEEE)*, pages 115–120, September 2013. doi:10.1109/ICeLeTE.2013.6644358.

8 Muhammad Ali Mazidi, Sarmad Naimi, Sepehr Naimi, and Shujen Chen. *ARM Assembly Language Programming & Architecture (Volume 1)*. MicroDigitalEd.com, 2013.

9 MicroDigitalEd. ARM Assembly Programming Using Raspberry Pi GUI, 2017. URL: `https://bit.ly/2JrkJaK`.

10 Robert G. Plantz. *Introduction to Computer Organization: ARM Assembly Language Using the Raspberry Pi*. Online, 2018. URL: `http://bob.cs.sonoma.edu/IntroCompOrg-RPi/intro-co-rpi.html`.

11 Andrew S. Tanenbaum and Todd Austin. *Structured Computer Organization*. Pearson Prentice-Hall, New Jersey 07458, 6 edition, 2013. URL: `https://goo.gl/N2YQc3`.

12 Hamid S. Timorabadi. Reduced complexity processor for teaching computer architecture. In *Proceedings of the Canadian Engineering Education Association (CEEA) Conference June 3-6, 2018 Vancouver BC*, 2018.