International Journal of Engineering Education Vol. 34, No. 3, pp. 1138–1144, 2018
Printed in Great Britain
0949-149X/91 $3.00+0.00

Comparing the Effectiveness of a Microprocessor Training System and a Simulator to Teach Computer Architecture*

NURETTIN TOPALOĞLU
Department of Computer Engineering, Gazi University, Ankara, Turkey. E-mail: nurettin@gazi.edu.tr

SAMI SAHIN
Department of Computer Education and Instructional Technology, Gazi University, Ankara, Turkey. E-mail: sami@gazi.edu.tr

ÇELEBI ULUYOL
Department of Computer Education and Instructional Technology, Gazi University, Ankara, Turkey. E-mail: celebi@gazi.edu.tr

Computer architecture is one of the basic courses in computer engineering departments. In this course both training systems and simulators are widely used for practice. The objective of this study is to compare a commercially available microprocessor training system (Lab-Volt) and a more affordable simulator (Visual 6502 Microprocessor Simulator). The comparison was based on 93 students enrolled in a computer architecture course. They were divided into two groups: one used the commercially available trainer and the other used the simulator. The assessment data were based on the results from a questionnaire completed by students and on their academic performance in the course. The results revealed that both tools support learning computer architecture and do not have significant differences in terms of learning outcomes. However, the simulator offers visual advantages compared to the training system.

Keywords: computer architecture; training system; Lab-Volt; simulator

1. Introduction

Computer architecture, which is mainly based on the working principles of microprocessors, is taught in computer engineering and electrical-electronics engineering departments. Computer architecture knowledge is as important as software and networking in computer teaching [1, 2]. The teaching of this domain consists of theoretical and practical parts. Teaching and learning activities are usually performed in the laboratory using a microprocessor training system and a simulator [3–8]. Students learn how to monitor a microprocessor based systems and perform transaction controls in these physical and virtual environments.

In the computer engineering department at Gazi University in Turkey where this research is conducted, the computer architecture course consists of 5 lessons each of 50 minutes. The curriculum includes microprocessor architecture, memory structure, input-output (I/O) techniques and principles of assembly language. The first three hours of the course are theoretical and the last two hours are the practical. The theoretical part of the course is based on the course book of Microprocessors and Assembly Language [9]. After the theoretical lectures, the students are able to perform coding using a microprocessor training system named Lab-Volt and view the results. A microprocessor simulator, Visual 6502, was also developed consistent with training system and the course book. By the use of simulator, the students compile the programs, run them at the desired speed, view the machine codes and data, trace the movements in the program, data and stack memory and watch how they move on the units of the computer architecture.

As a result of using Lab-Volt and Visual 6502, students learn the following content according to the objectives of Computer Engineering Curricula, Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering, CE2016 ACM/IEEE [1].

- Bits, bytes, and words.
- Numeric data representation and number bases.
- Basic organization of the von Neumann machine.
- Control unit; instruction fetch, decode, and execution.
- Control unit: hardwired realization vs. microprogrammed realization.
- Instruction sets and types (data manipulation, control, I/O).
- Assembly/machine language programming.
- Instruction formats.
- Addressing modes.
- Subroutine call and return mechanism.
- I/O fundamentals and interrupts.
- Main memory organization and operations.

When the literature is examined, there are academic studies in which both Lab-Volt and simulators used independently in educational environments [10–12]. The comparison of these two tools reveals the opportunity for which tool to be more effective in which educational variables. In this study, these two

* Accepted 10 February 2018.

training tools were examined in terms of various variables such as ease of navigation, design, accessibility, ease of use and productivity. This comparison is quite useful for educators who use them in the courses to see the advantages and disadvantages of these tools.

2. Hardware and software teaching tools

2.1 Lab-Volt training system

Lab-Volt [13, 14] is used to teach the essentials of computer systems (Fig. 1). Lab-Volt consists of five parts: microprocessor, memory (RAM and ROM and EPROM programming socket), I/O unit and error generating unit. Students are required to compile their programs according to the instructions. When a program is run that targets the I/O unit, the results are seen on six 7-segment displays, four of which are addresses and the other two of which are data.

The hex keypad has a second function key that can be selected during operation to view the contents of the registers and memory as real time. During the running of the codes, a number of errors can be made with the DIP micro switches on the training system. This helps the student solve the mistakes.

2.2 Visual 6502 microprocessor simulator

Visual 6502 [15] was developed as an alternative to Lab-Volt training system (Fig. 2) and removed some blocks of creativity in teaching engineering [16]. It is composed of local editor, assembler, debugger, animator and virtual I/O. Visual 6502 gives students the opportunity to write programs in assembly language, compile them and observe the results. It helps to operate events in computer architecture either step-by-step at a specific speed or at normal speed. Students can correct syntax and logical errors during debugging using a breakpoint. While the simulator runs programs in the real time, the debugging can be viewed on a single Integrated Development Editor (IDE). Momentary contents of registers can be seen as binary, decimal, hexadecimal and ASCII, and flags can be seen as bit-based. In the memory segment, program, data, stack and contents can be tracked along with their addresses and contents, helping the process. If the operated program contains I/O operations, the results of this process can be seen on virtual LEDs, micro switches, traffic lights and LED indicators.

The most important issue in computer architecture education is related to what is going on in the system. It is important to see how program codes affect hardware components. The simulator has a mechanism that instantly simulates the events that occur in computer system (Fig. 3). This integrated structure includes all hidden and visible microprocessor registers, control unit, memory, I/O unit, communication paths, operated program codes and control panel.

Students spend half of the two-hour laboratory in the training system and the other half in the microprocessor simulator. In the first hour, they

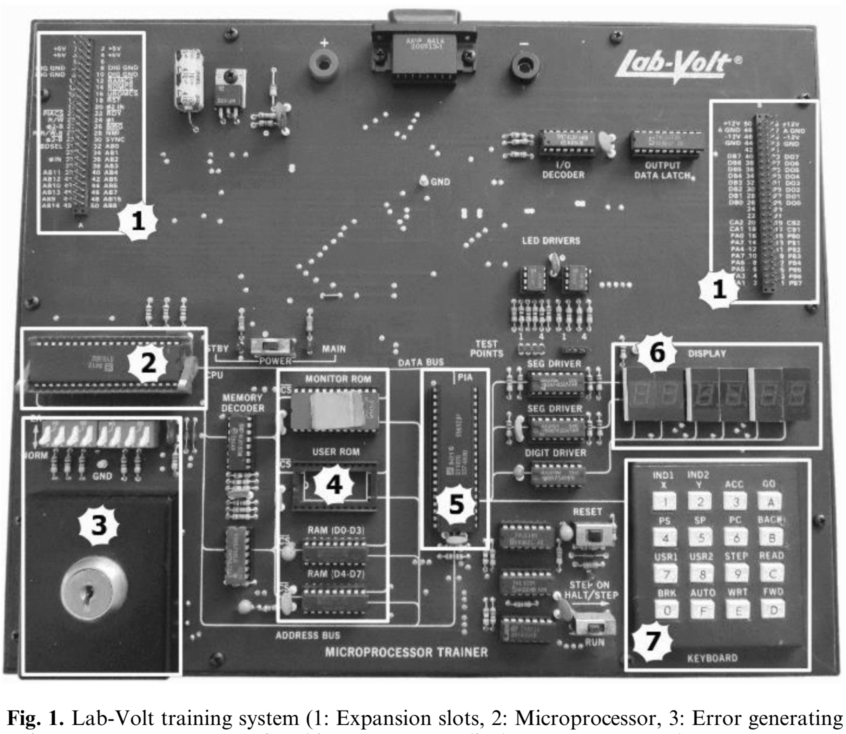

Fig. 1. Lab-Volt training system (1: Expansion slots, 2: Microprocessor, 3: Error generating units, 4: Memory group, 5: I/O Chip, 6: 7-segment displays, 7: Hex Keypad).

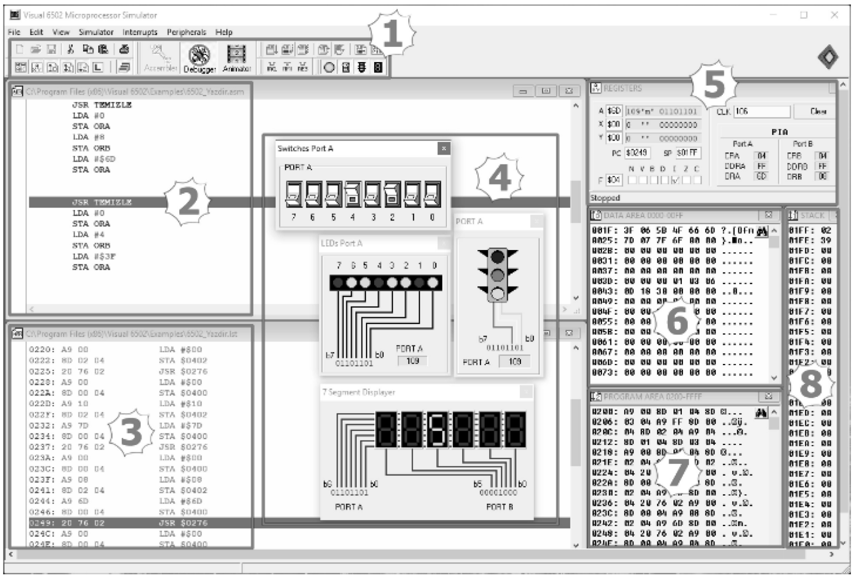

Fig. 2. Visual 6502 microprocessor simulator (1: Toolbox, 2: Local editor area, 3: Assembled program area, 4: Virtual I/O elements, 5: Registers' area, 6: Program memory area, 7: Data memory area, 8: Stack memory area).

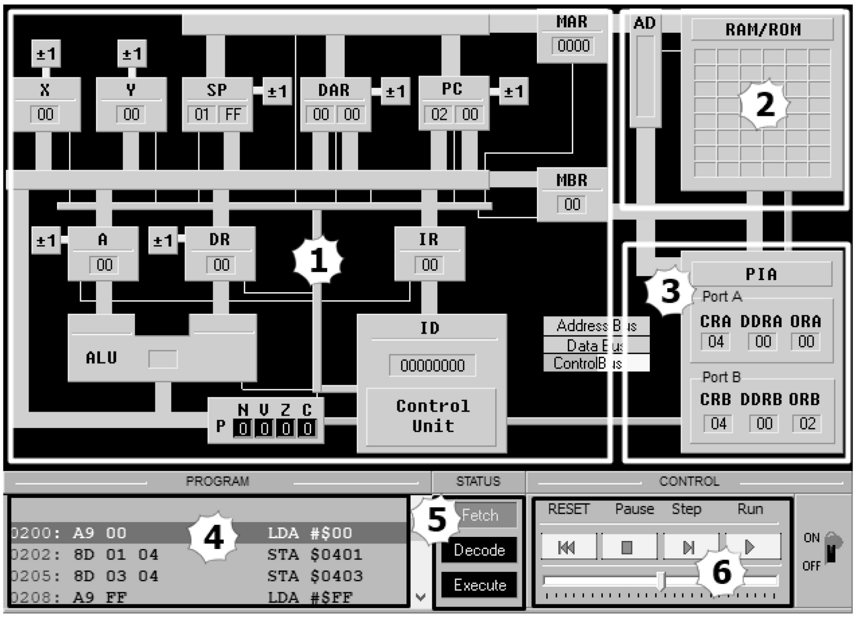

Fig. 3. Visual 6502 animators (1: Processor units, 2: Memory, 3: I/O unit, 4: Program area, 5: Process stages, 6: Control buttons).

observe the outputs and the codes of the programs they write. Using the I/O ports on the training system, many devices on the outside world (special led arrays and traffic lights etc.) can be controlled by using logic signals. In the last hour, they write the same programs in the simulator editor, and observe the effects visually and also compare and interpret the results. The application is done with the PCs that are available in the lab or laptop that the students bring with them. Table 1 shows advantages and disadvantages of Lab-Volt and Visual 6502.

The purpose of this study was to compare the usability of Lab-Volt and Visual 6502 used in the laboratory of the computer architecture course. In addition, the academic success of the students was compared to examine the effects of these tools on student success.

3. Method

This section provides explanations of the research process, data collection tools and experimental groups.

Table 1. Comparison of Lab-Volt and Visual 6502 by categorical

Category	Lab-Volt Training System	Visual 6502 Microprocessor Simulator
Cost	The cost is high. It depends on the functionality and features. The cost per student will increase when the crowded classes are considered.	The cost is inexpensive; many free or cheap simulators can be obtained.
Visualization	Microprocessor operation and developing events cannot be observed clearly. Functioning and instruction effects cannot be noticed visually.	The events within the processor and functioning are completely visualized and the effects of the instruction on the elements is noticeable.
Malfunction Risk	It is consisting of electronic and mechanical parts, so it has high risk of breakdown. It requires maintenance and repair.	There is no risk of physical malfunction.
Error check	It is difficult and time consuming to enter program codes when it is not integrated in PC. Backtracking is tedious when there is incorrect code entry.	Errors are immediately corrected.
Observation	It is difficult to understand and follow the schedule of the program because each machine translation cannot be tracked due to it lacks visibility.	It is easy to follow, events that develop can be observed visually in real time.
Limitation	Established in the laboratory. Due to limited laboratory hours, there is limited use time.	It can be installed and used on a PC and a Notebook.
Editor and Compiler	No editor or compiler.	Editor and compiler built in.

3.1 Implementation process

The computer architecture course is a 16-week course. The first four weeks of the course covers theoretical base of the computer architecture and the remaining weeks covers also practical units. Computer architecture courses are held on Tuesday and Thursday in 2 branches for 3 hours theoretical and 2 hours practical. In the theoretical part of the course, the functions of the elements bringing up the computer architecture and how they are audited are explained with the help of the ''Assembly'' programming language and then the implementations are carried out in the microprocessor laboratory. The students in both classes were divided into two groups (four groups in total), while the implementation of one half made carried out with the simulator, the other half used the training system. Students of both groups observed the effects of the program codes belonging to the topics taught in the theoretical courses on the hardware elements.

At the end of a three-hour theoretical course, after a 15-minute break, the students go to the laboratory to apply the programs for the subjects that they are working on and test their programs on the Lab-Volt training system in the first place. Firstly, the theoretically written program/programs are compiled via the handy assembler and the program codes are entered into the training system memory via the hex keypad. Then, the program is run, and the results are displayed on six 7-segment displays. The effects and reflections of the program codes entered into the experiment system on the 7-segment LEDs can be observed step-by-step by taking advantage of the debugging feature of the training system. In the second part of the lab hours, the same programs are implemented on the Visual 6502 simulator installed in the lab. The students can write and compile the program in the simulator editor first and can correct it immediately if there is any mistake. Then the debugger is run, and it recovers the program from logic errors. At this stage, the student can make as many changes as he or she desires on the program and sees the results on the registers and the data changes in memory. Later, through the animator, the student can observe how the program he/she wrote performs real-time and controls the hardware components. Finally, students can watch how the programs they wrote effect LEDs, micro-switches, traffic lights and 7-segment displays.

3.2 Data collection tools

A questionnaire consisting of 20 items was prepared for data collection in this research [17]. In the preparation of the questionnaires, the literature on the use and the benefits of the training systems were examined. This questionnaire, which was prepared by researchers, contains several questions about the use of both Lab-Volt training system and Visual 6502 microprocessor simulator.

The achievement tests were prepared by the faculty member who teach the course taking the learning outcomes into consideration. The questions were prepared to cover the whole of the topics taught in the courses. In the achievement test, open-ended questions were avoided as much as possible. One midterm exam and one final exam were prepared for the course. In the midterm exam, there are some questions from the subjects in the first 8 weeks. In the final exam, there are questions in the whole semester (16 weeks) including midterm topics. Two of the five questions were chosen from theoretical topics, and the remaining three were selected from the topics related to programming and practice. The programming questions were designed to test the achievements of the students in the laboratory. The test questions were prepared and evaluated in the format of ECTS (European Credit Transfer and Accumulation System) [18].

3.3 Procedure

64 male and 29 female students participated in the research. The treatments were carried out in 4 different branches of computer architecture course taught by the same faculty member. In all 4 branches the number of students was between 20 and 25. Two of these branches were on Tuesday and the other two were on Thursday. One of the two branches of these days used training system and the other branches used microprocessor simulator.

4. Findings

When the mean scores evaluated, it is seen that the students expressed positive opinions in general in majority of the items in both groups (Table 2). However, for item 5 (I have enough supportive documents (books, documents, test charts and help section that I can take advantage of.) students in both groups were indecisive. While only the training system group's students were indecisive for items 6, 13 and 17; only the simulator group students were indecisive for item 4. Moreover, the simulator group's students chose "strongly agree" option for items 1, 2, 3, 8 and 11, the training system group chose "strongly agree" option only for item 1. it is seen that the students of the simulator group prefer "strongly agree" option for "buttons, keys, switches and similar elements are in a proper place and easily accessible.", "It is understandable what buttons, keys, switches and similar elements mean.", "the colors and the text on the training system facilitate the work.", "It helps me to learn the subjects." and "the program allows you to write, correct, run and debug" expressions. The students of the training system group chose "agree" option for these expressions. While the simulator group was "indecisive" for the "It works fine" expression, the training group chose "agree". Besides, the students of the training group were "indecisive" for "It is user friendly and suitable for self-learning", "It allows me to see the operation phases of the program (fetch-decode-execute) with the help of graphical interfaces." and "It is easy to control any element in the system in the training system" expressions. On the other hand, the simulator group preferred "agree" choice for these expressions. In addition, both groups were indecisive about "I have enough supportive documents (books, documents, test charts and help section) that I can take advantage of." expression.

The mean scores that give agreement degrees of the students on the items are examined in Table 2 and it appears that there are differences among the groups. Whether these differences were statistically significant was analyzed by independent sample t-test and the results are presented in Table 3. Accordingly, the perception of training system is significantly higher than the simulator in terms of "the system's smooth operation". In other respects; the simulator group has a significantly more positive perception in terms of "markers like writing and coloring make it easier to work"; "User friendly and suitable for self-learning"; "Writing, correcting, running and debugging"; "ensuring that the program's working phases can be viewed with the help of graphical interfaces" and "facilitating the supervision of any component". When the findings are evaluated in general, it is seen that the simulator application is perceived more positively than the training system.

In terms of academic success, the mean and standard deviations of the experimental groups are shown in Table 4. It is seen that the groups have very close scores regarding the midterm and final exam and the achievements are in moderate level. The statistical significance of the differences between the means was examined with independent samples t-test and no significant difference was found between the scores of the groups. When standard deviations are considered, scores of the students in the final exam becomes more heterogeneous than the scores of the midterm exam.

5. Discussion and conclusion

Although microprocessor training systems are still the main teaching tools in computer architecture courses [10], simulators have also alternatively been used in the recent years. As a result of the improvements in web and software technologies and the progress in the speed of the internet, the use of simulators has increased in the training environments. In this study, a training system and a simulator were compared in teaching computer architecture to determine their relative advantages. The results showed that the students more perceived the simulator as easy to understand and use. They also perceived the colorization and explanatory texts of the simulator more helpful. The students were better able to control the programming via the graphical interface of the simulator. Thus, the simulator was found more successful, more user-friendly and more suitable for learning because of its visual features. On the other hand, while the students using the training systems had found it run-

Table 2. Mean and standard deviations by experimental groups

Criteria: 2.61 -3.40: indecisive (Und); 3.41-4.20: Agree (A); 4.21-5,00: Strongly Agree (SA)

	Group	M	Sd	Participation
Buttons, keys and similar elements are in a proper place and easily accessible.	Lab-Volt Visual	4.26 4.28	0.88 0.76	SA SA
It is understandable what buttons, keys, switches and similar elements mean.	Lab-Volt Visual	4 4.24	0.82 0.87	A SA
The colors and the text on the training system facilitate the work.	Lab-Volt Visual	3.7 4.43	1.12 0.76	A SA
It works fine.	Lab-Volt Visual	3.67 2.82	1.29 1.02	A Und
I have enough supportive documents (books, documents, test charts and help section) that I can take advantage of.	Lab-Volt Visual	3.3 2.98	1.19 0.94	Und Und
It is user friendly and suitable for self-learning.	Lab-Volt Visual	3.07 3.54	1.2 0.93	Und A
It needs to be improved.	Lab-Volt Visual	3.98 4.16	1.1 0.89	A A
It helps me to learn the subjects.	Lab-Volt Visual	4 4.34	0.98 0.75	A SA
I hope that what I learn from the applications will be useful for my professional work in the future.	Lab-Volt Visual	3.53 3.72	1.01 0.93	A A
It offers the opportunity to interactively view registers, processor, memory and input-output systems.	Lab-Volt Visual	3.86 4.18	0.99 0.69	A A
The program allows you to write, correct, run and debug.	Lab-Volt Visual	3.72 4.28	1.16 0.76	A SA
It shows the effect on the hardware of programs written in ''Assembly'' programming language.	Lab-Volt Visual	4.19 4.14	0.82 0.64	A A
It allows me to see the operation phases of the program (fetch-decode-execute) with the help of graphical interfaces.	Lab-Volt Visual	3.07 4.08	1.28 0.88	Und A
The practical exercises teach me how a microprocessor system works.	Lab-Volt Visual	4 4	0.82 0.76	A A
It teaches the elements, units and functions of computer architecture.	Lab-Volt Visual	3.7 3.71	0.89 0.84	A A
It is encouraging to write and develop programs.	Lab-Volt Visual	3.51 3.82	1.2 0.96	A A
It is easy to control any element in the system in the training system.	Lab-Volt Visual	3.12 3.74	1 0.85	Und A
It provides effective participation in learning.	Lab-Volt Visual	3.77 4.14	0.97 0.81	A A
It makes the learning process entertaining and engaging.	Lab-Volt Visual	3.98 4.02	1.12 0.91	A A
It increases the interest in the course and provides concentration.	Lab-Volt Visual	3.81 3.9	1.07 1.04	A A
Overall	Lab-Volt Visual	3.71 3.93	0.66 0.43	A A

Table 3. Independent samples t-test results

	Lab-Volt (n = 43)		**Visual (n = 50)**				
	M	**Sd**	**M**	**Sd**	**t**	**df**	**p**
The colors and the text on the training system facilitate my work.	3.70	1.12	4.43	0.76	3.69	90	0.000
It works fine.	3.67	1.29	2.82	1.02	3.57	91	0.001
It is user friendly and suitable for self-learning.	3.07	1.20	3.54	0.93	2.12	91	0.036
It allows you to write, correct, run and debug.	3.72	1.16	4.28	0.76	2.79	91	0.006
It allows me to see the operation phases of the program with the help of graphical interfaces.	3.07	1.28	4.08	0.88	4.49	91	0.000
Controlling of any element is easy.	3.12	1.01	3.74	0.85	3.24	91	0.002

Table 4. Mean scores of the mid-term and final exam

	Test	N	M	Sd
Midterm	Lab-Volt	40	45.20	16.36
	Visual	44	42.45	15.40
Final Exam	Lab-Volt	40	39.55	21.86
	Visual	38	42.08	24.97

ning smoothly, the students using the simulator were indecisive in this regard. The studies in the literature demonstrated similar results of the simulators in the different subject areas [2, 4, 10, 19–21]. Consequently, we recommend the use of both tools in computer architecture programming courses since both have relative advantages and create no significant difference in the achievement.

References

1. Computer Science Curricula, Curriculum Guidelines Undergraduate Degree Programs in Computer Science-Association for Computing Machinery, ACM/IEEE Computer Society, December 20, 2013.
2. F. Sanchez, D. Megias and J. Prieto-Blazquez, SIMR: A Simulator for Learning Computer Architecture, *International Journal of Engineering Education*, **27**(2), 2011, pp. 238–247.
3. Hala ElAarag, Teaching Computer Organization: A Practical Approach, *Journal of Computing Sciences in Colleges*, **28**(2), 2012, pp. 210–217.
4. Dale Skrien, CPU Sim 3.1: A Tool for Simulating Computer Architectures for Computer Organization Classes, *ACM Journal of Educational Resources in Computing*, **1**(4), pp. 46–59, 2001.
5. P. W. C. Prasad, Abeer Alsadoon, Azam Beg and Antony Chan, Using Simulators for Teaching Computer Organization and Architecture, *Computer Applications in Engineering Education*, **24**(2), 2016, pp. 215–224.
6. S. Z. Sweidan and K. A. Darabkh, A new efficient assembly language teaching aid for intel processors, *Computer Applications in Engineering Education*, **23**(2), 2015, pp. 217–238.
7. D. Patti, A. Spadaccini, M. Palesi, F. Fazzino and V. Catania, Supporting Undergraduate Computer Architecture Students Using a Visual MIPS64 CPU Simulator, *IEEE Transactions on Education*, **55**(3), 2012, pp. 406–411.
8. E. A. QaRalleh and K. A. Darabkh, A New Method for Teaching Microprocessors Course Using Emulation, *Journal of Computer Applications in Engineering Education*, **23**(3), 2015, pp. 455–463.
9. N. Topaloğlu, *Mikroişlemciler ve Assembly Dili*, Turkish, 6th Edition, Seckin Publishing, Ankara, 2015.
10. N. Topaloglu and O. Gurdal, A Highly Interactive PC Based Simulator Tool for Teaching Microprocessor Architectures and Assembly Language Programming, *Electronics and Electrical Engineering*, **2**(98), 2010, pp. 53–58.
11. E. Avci, İ. Turkoglu and M. Poyraz, Intelligent target recognition based on wavelet adaptive network based fuzzy inference system, *Expert Systems with Applications*, **29**(1), 2005, pp. 175–182.
12. N. Jovanovic, Z. Jovanovic and A. Jevremovic, Evaluation of Simulators for Teaching Computer Networks, *International Journal of Engineering Education*, **32**(5), 2016, pp. 2098–2106 Part A.
13. Lab-Volt by Buck Engineering Co. Inc., New Jersey, USA, 2009.
14. Lab-Volt, Microprocessor Concepts and Applications, 1st Edition, 14th Printing, Buck Engineering, USA, 1996.
15. Visual 6502 Microprocessor Simulator, http://w3.gazi.edu.tr/nurettin/visual_en.html, accessed 05 March 2018.
16. Z. Liu and D. J. Schönwetter, Teaching Creativity in Engineering, *International Journal of Engineering Education*, **20**(5), 2004, pp. 801–808.
17. Data collection tools: survey instruments, http://w3.gazi.edu.tr/nurettin/survey_IJEE.pdf, 10 September 2017.
18. European Credit Transfer and Accumulation System, http://ec.europa.eu/education/resources/european-credit-transfer-accumulation-system_en, 05 March 2018.
19. S. Görgünoğlu, M. Peker, B. Şen and A. Çavuşoğlu, An Efficient Pseudo Microprocessor for Engineering Education, *Procedia Technology*, **1**, 2012, pp. 36–43.
20. B. Hatfield, M. Rieker and L. Jin, Incorporating simulation and implementation into teaching computer organization and architecture, *ASEE/IEEE Frontier in Education Conference*, Indianapolis, USA, 2015, pp. 19–22.
21. B. Naz, S. Hyder, M. H. Unar and S. Naveed, Design & Development of A Simulator for Microprocessor Application Trainer (MATSOFT385), *International Journal of Engineering Works*, **1**(2), 2014, pp. 45–47.

Nurettin Topaloğlu is currently an Associate Professor of Computer Engineering Department of Technology Faculty at Gazi University in Turkey. He received BSc in Electronics, MSc in Electronics and Computer Education and PhD in Electric Education. His research interests are computer architecture and organization, informatics technologies and information security. He has been involved in research areas in deep and machine learning. He developed the educational software Visual 6502 Microprocessor Simulator for teaching computer architecture. He is a writer of Microprocessors and Assembly Language, and x86 Microprocessor Architecture and Assembly Language in Turkish.

Sami Şahin is a faculty member of Education Faculty of Gazi University in Turkey. His main interest is utilizing emerging technologies into teaching and learning. Dr. Şahin has BS in Physics; MS in Instructional Design, Development and Evaluation; and PhD in Curriculum and Instruction with specialization in instructional technology. Dr. Şahin has published several articles, books and book chapters in the field of instructional technology. His recent publications have been on preservice teacher's technology use.

Çelebi Uluyol is a PhD and has been working at the Department of Computer Education and Instructional Technology at Gazi University in Turkey. He has published many scientific articles in national and international journals about technology integration in education. Especially in recent years, he is working about augmented reality learning and making experimental studies related to augmented reality in educational settings. He continues to teach graduate courses related to the analysis and design of augmented reality. He is also planning to study personal learning environments in education within the next five years. Personal learning environments (PLEs) are Web 2.0 tools and services by which users access, construct, manage, and share educational contents in order to meet their learning needs.