International Journal of Computer Applications (0975 – 8887)
Volume 180 – No.47, June 2018

Using two Educational Simulator Tools for Computer Architecture Teaching and Learning Support

Dimitris Kehagias
Department of Informatics
T.E.I. of Athens
Greece

ABSTRACT

Simulators are commonly used in any computer architecture course as primary tools for supporting the teaching and learning activity. We have developed two educational simulator tools to support teaching and learning of the MESI cache coherence protocol and dynamic scheduling using Tomasulo's Algorithm.

We have used these simulators during the spring semester of the academic year 2016 – 2017, in the context of the “Advanced Computer Architecture” course offered by the Informatics department of the Technological Educational Institute (T.E.I.) of Athens.

In this paper we briefly present these simulators and evaluate their impact on the learning process. The results are presented both qualitatively and quantitatively and are strongly indicate that the use of the two simulators can effectively support the learning process and enhance learning.

General Terms

Computer Simulation, Algorithms, Applied Computing.

Keywords

Tomasulo’s algorithm, MESI protocol, Simulator, Computer architecture, Interactive animation.

1. INTRODUCTION

For computer science students in an undergraduate advanced computer architecture course, the topics of cache coherence and dynamic scheduling are often confusing as they are not always that distinct. In order to help students learning these course topics and engage their learning interest, simulation tools have been created and used in the past. We, in the department of Informatics of the Technological Educational Institute (T.E.I.) of Athens [1], in the content of the undergraduate course “advanced computer architecture”, created two simulators and employ them as the primary tools for explaining the aforementioned computer architecture topics. These simulators are a MESI simulator for a write-back cache [2] and an android based simulator that shows how dynamic scheduling is obtained using Tomasulo's Algorithm [3].

Our intention to build these simulators was motivated by the fact that many students, taking the “advanced computer architecture” course offered by our department, exhibit difficulties fully understand (a) the cache coherency problem and (b) how dynamic scheduling is obtained using Tomasulo's Algorithm.

The aim of this paper is to investigate whether the teaching of the MESI cache coherence protocol and dynamic scheduling using Tomasulos’ algorithm can be made more effective through the use of the two simulators as we have built them. The evaluation process uses pre and post-testing and questionnaire analysis.

The rest of the paper is organized as follows. Section II provides a brief overview on the most relevant educational simulators. Section III explains the features of the two simulators as we have built them. Section IV demonstrates the evaluation of the two simulators. Section V concludes the paper.

2. RELATED WORK

The evaluation of the simulators was carried out using both qualitative and quantitative methods. Similar evaluations we found in [4, 5, 6, 7].

Various standalone tools exist to explain how dynamic scheduling is obtained using the Tomasulo's Algorithm, and to simulate the MESI protocol. The most relevant ones are presented in the following lines:

In [8] a HASE simulation model, which closely follows the design of the IBM system 360/91 floating-point unit, has been built in order to demonstrate dynamically the Tomasulo's algorithm.

The simulator in [9] simulates Tomasulo's algorithm for a floating-point MIPS-like instruction pipeline, demonstrating out-of-order execution.

[10] and [11] present two web-based tools that have been developed for students to understand the concepts of the Tomasulo's algorithm used for dynamic scheduling.

The simulator in [12] simulates a run of a software application in a cached multiprocessor system and uses the MESI protocol to maintain data coherence.

In [13] a flash interactive animation shows how MSI and MESI cache coherence protocols work.

However, none of them includes all the features that our proposal offers. These features include operation in a step by step mode, animation, written explanations in every animation step, configurable execution core, variable issue rate, variable latency per instruction class. Also, allows the user (i) to see memory contents during simulation, (ii) to show or hide animations and get help during simulation.

3. THE TWO SIMULATORS

3.1 The MESI cache coherence simulator

Figure 1 shows the graphical interface of the simulator whose features include:

- *Three cores with local caches*. The caches are direct mapped with a write back policy. The local cache of each core has four cache lines (LN0-LN3) and the cache-line/block size is four words. The column titled STATE displays the current state (M, E, S, or I) of each cache line. The simulator doesn't concern itself with byte addressing within words, word alignment and so on. The simulator accepts its input from users. A user specifies a word in the "Enter Word…." frame and starts a read/write transaction on the specified word by pressing the READ/WRITE button. A word consists of a letter (A-Z) and a digit (0-9). In order to start a read/write operation the contents of memory must previously be set by pressing the RND Words button. The words are set randomly.

- *A main memory* containing sixty four words organized as sixteen memory blocks (BL0-BL15) with four words each. Memory is addressed at block level and data addresses start from zero. Thus, address zero indexes the first block in memory, address one the second block, and so on.
- The local caches and the main memory are connected by *a bus* that acts as a communication network.
- *LOG info panel* that shows which read or write request is being executed.
- With the *Initialize button* the contents of all caches and main memory are cleared, while with the *RND Words button* the contents of memory are specified randomly.
- The simulator permits simulated execution to proceed in variable-speed timed mode with interactive display update speed adjustment using the *frame Speed: (5 to 100).* Speed of 10, 50 and 100 correspond to 10, 2 and 1 second(s) per step respectively.
- By selecting the *Scenario button* twelve ready case studies are displayed that implement all the functioning parts of the MESI protocol.
- Description of the main interface is given by selecting the *Help button.*
- *Main menu button* returns to main menu.

3.2 The Tomasulo simulator

The simulation screen (Figure 2) has a very rich and friendly visual interface. It illustrates the movement of instructions to the reservation stations and the movement of results from the functional units. It consists with the following components:

RAT: Register Alias Table is a structure for performing register renaming. It maintains the mappings between reservation stations and destination registers of instructions.

LOAD Q / STORE Q: Load and store buffers for LD and SD instructions. They hold data and addresses for memory access.

INST Q: The "INST Q" component is a queue that contains the instructions in the order entered by the user. The instructions are issued into the reservation stations in first-in, first-out order.

REGS: The "REGS" component implements the Floating-point (F) and integer (R) register file. The registers contain values entered by the user during the configuration process, or broadcasted since instructions complete their execution. These values that are already in registers, meaning the values that are present and ready for execution, are entered to reservation stations.

ADD RS / MUL RS: There are two types of reservation stations "ADD RS" and "MUL RS". One is for ADDD and SUBD instructions, while the second is for MULTD and DIVD instructions. Each reservation station is made up of three fields. The first field in a row holds the opcode for the pending instruction in the form of an arithmetic symbol (+,-,*,/, for ADDD, SUBD, MULTD and DIVD instructions respectively) and the other two fields hold either operand values, or names of reservation stations or load/store buffers that will provide them.

ALU ADD / ALU MUL: Functional Units (FUs) to accomplish the execution step of instructions. The "ALU ADD" FUs are floating point adders which execute ADDD and SUBD instructions while the "ALU MUL" is floating point multipliers which execute MULTD and DIVD instructions. The FUs receive instruction and operand packets from the RSs and send operand result packets to the common data bus. The number of clock cycles required to execute an instruction is a parameter read from the hardware configuration activity at the start of a simulation.

All the above mentioned components are interconnected with a common data bus (CDB), which is used to broadcast result from the adder, multiplier and the load buffer to the reservation stations, the register file and the store buffers.

The simulation screen provides the user with several choices, including:

ISSUE: During the issue process the next -in program order- instruction is taken from the instruction queue and putted into a free reservation station of correct kind (ADD RS or MUL RS).

DISPATCH: The process of sending an instruction to execution from a reservation station to a functional unit (ADD RS to ALU ADD or MUL RS to ALU MUL).

EXECUTE: Is the phase during which a functional unit (ALU ADD or ALU MUL) operates on ready operands of an instruction.

BROADCAST: When an instruction finishes execution broadcasts its results on a common data bus and from there into registers and reservation stations.

NEXT EVENT: Allows the user to move to the cycle in which some visible action occurs.

MEMORY CONTENTS: Memory contents can be seen during simulation. ANIMS: Show or hide animations.

4. EVALUATION

In this paper we have investigated on a first level whether the teaching of MESI cache coherence protocol and dynamic scheduling using Tomasulos' algorithm can be made more effective through the use of the two simulators as we have built them. Thus, the questions raised are as follows:

- Did these learning simulator tools benefit students and helped them understand the topics under investigation?
- What was the degree of usability of the two simulators?

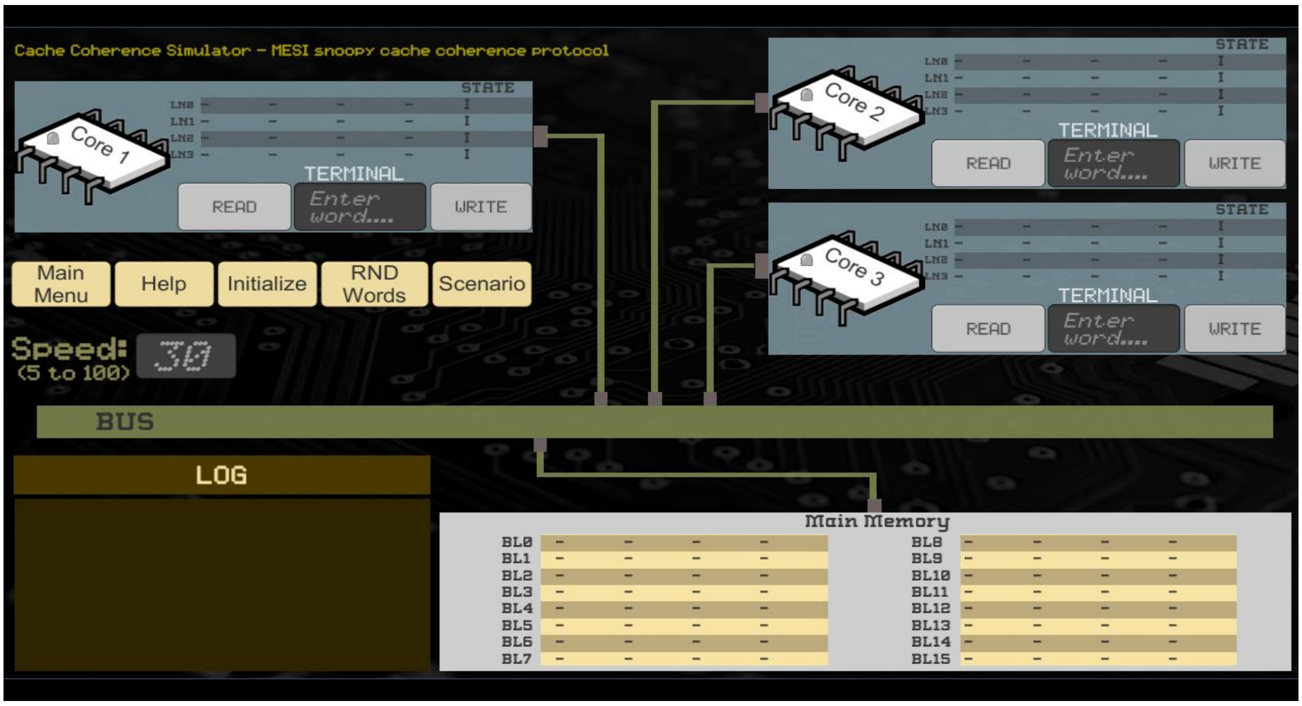

Figure 1 [2]: MESI simulator user interface

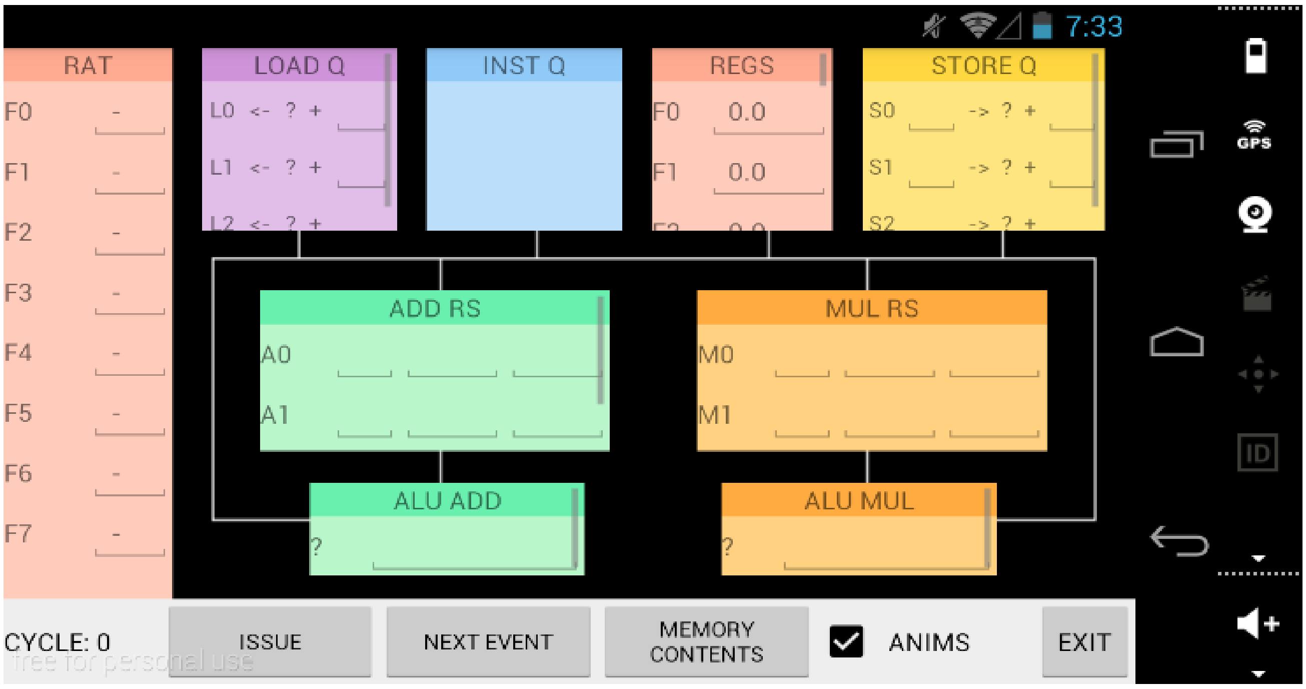

Figure 2 [3]: Tomasulo’s simulation screen

These questions were addressed through a combination of quantitative and qualitative methods. The qualitative method used opinion surveys, while the quantitative one pre and post-tests.

4.1 Methodology

The evaluation of the two simulators was carried out in the framework of the undergraduate “Advanced Computer Architecture” course at the department of Informatics of the Technological Educational Institute (T.E.I.) of Athens during the spring semester of the academic year 2016 - 2017. This course is taught during the sixth semester.

During the semester, students, having been taught about MESI cache coherence protocol and the Tomasulo algorithm, wrote a pretest on these two topics to determine the extent of understanding the issues. The pre-test had the form of a midterm examination. Students' answers revealed that 20 students out of the 48 students faced difficulties in understanding the concepts as taught in the traditional way. After that, the group of the 20 students was used as the test group. Initially, the group of the 20 students performed a number of tutorial exercises, addressing the identified learning difficulties, using the two simulators. After two weeks of this educational process, the students participated in a post-test to determine whether the use of simulators helped them to understand better the topics under investigation; the post-test had the form of a second midterm for the test group, consisted of questions similar to the pre-test questions. They also answered to an opinion survey regarding the two simulators they used.

4.2 Results

4.2.1 Qualitative evaluation

One opinion survey was conducted at the end of the evaluation phase. The survey was implemented using 5-point Likert scale questionnaire (5=strongly agree, 4=agree, 3=don't know, 2=disagree, 1=strongly disagree) and consisted of eight opinion questions. Questions 1 to 4 (Q1-Q4) investigate the extent to which the two simulators were effective in helping students to understand specific course topics.

The questions concerning the Tomasulo simulator were the following:

- Q1. The simulator helped me understand how each stage of Tomasulo's algorithm operates (issue, execute and write back).
- Q2. I understood better the way instructions are completed out of order using the simulator.
- Q3. The simulator helped me understand the issue of how register renaming is provided by reservation stations.
- Q4. The simulator helped me understand the issue of how Tomasulo's algorithm eliminates WAW/WAR hazards.

The questions concerning the MESI simulator were the following:

- Q1. The simulator helped me understand the meaning of each state of the MESI protocol in the cache memory and the corresponding state in any other cache memory in a multiprocessor environment.
- Q2. The simulator gave me the opportunity to study in depth the cache memory parameters, such as the associativity, the replacement policy, the writing policy, etc.
- Q3. Using the simulator I better realized when a transition between states is needed in order to reflect the actions taken by a processor.
- Q4. The simulator helped me understand program locality.

The results are shown in Tables I and II.

Questions 5 to 8 (Q5-Q8) are common for both simulators. They gather information about the effect of the simulators on the students', overall, learning process and the user interface. These questions were the following:

- Q5. The use of the simulator facilitated better understanding the theoretical background of some of the difficult concepts of the course.
- Q6. The use of the simulator has increased my confidence to do well in the course.
- Q7. I found the user interface of the simulator easy to understand and use.
- Q8. The use of the simulator helped me to achieve the course learning outcomes.

The results are shown in Tables I and II.

Table 1. Tomasulo simulator

Question	1	2	3	4	5	Average grade
Q1		1		10	9	**4.35**
Q2			1	5	14	**4.65**
Q3		1	1	4	14	**4.55**
Q4	1	2	1	10	6	**3.90**
Q5			1	4	15	**4.70**
Q6		2	1	11	7	**4.30**
Q7			1	4	15	**4.70**
Q8		1		12	7	**4.25**

Table 2I. MESI simulator

Question	1	2	3	4	5	Average grade
Q1		2		11	7	**4.15**
Q2				7	13	**4.65**
Q3			2	12	6	**4.20**
Q4			3	12	5	**4.10**
Q5			1	4	15	**4.70**
Q6		2	1	10	8	**4.35**
Q7			1	4	15	**4.70**
Q8		1		13	6	**4.20**

As shown in Tables I and II, the results of the survey were positive, overall.

For both simulators, highest ratings were obtained for questions 5 (Q5), concerning the effectiveness of using the simulators towards helping students to understand specific course topics (average grade 4.70), and 7 (Q7), concerning the simplicity of the user interface (average grade 4.70). The overall helpfulness of the simulators is demonstrated by the high ratings of questions 5 to 8 (Q5-Q8).

The ratings for questions 1 to 4 (Q1-Q4) show the extent to which the two simulators were effective in helping students to understand specific difficult issues related to the MESI cache coherence protocol and dynamic scheduling using Tomasulo's Algorithm.

4.2.2 Quantitative evaluation

The quantitative method used pre and post-tests. After the midterm examination (multiple choice pre-test), 20 of the students, who faced difficulties in understanding the concepts as taught in the traditional way, constituted the test group and performed a number of tutorial exercises addressing the identified learning difficulties using the two simulators. At the end of this educational process the students participated in a multiple choice post-test to determine whether the use of simulators helped them to understand better the topics under investigation. The pre and post-tests consisted of 5 similar questions

Table III summarizes the results obtained from the pre and post-tests, and shows that the experience with both simulators seems to influence to a greater extent the topics under examination. This is evident in some of the individual post-test results.

Table 3II: Results from pre-post tests

Examining MESI cache coherence protocol (% correct answers):						
Test	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**	**Average grade**
Pre	45.2	37.3	49.5	27.4	20.6	**36**
Post	56.5	66.4	59.3	72.3	65.4	**64**
Examining dynamic scheduling using Tomasulo's Algorithm (% correct answers):						
Pre	38.4	39.3	22.4	48.2	28.4	**35.3**
Post	55.4	60.3	70.4	55.5	58.8	**60.1**

5. CONCLUSIONS

In this paper we have investigated on a first level whether the teaching and learning of MESI cache coherence protocol and dynamic scheduling using Tomasulos' algorithm can be made more effective through the use of two simulators as we have built them.

The impact of the simulators on the learning process was evaluated by means of an opinion survey as well as pre and post-tests given by students after the completion of the topics under investigation.

The results obtained have been encouraging, indicating that the use of the two simulators can effectively support the learning process and enhance learning.

6. REFERENCES

[1] "Advanced Computer Architecture". Available at: http://www.cs.teiath.gr/?page_id=6450.

[2] D. Kehagias and I. Raptis, "An Interactive MESI Cache Coherence Simulator for Educational Purposes", In the ACM Conference Proceedings of the 20th Pan-Hellenic conference on Informatics (PCI 2016), Patra Greece, doi>10.1145/3003733.3003765, Nov. 10-12, 2016.

[3] Dimitris Kehagias and V. Douskas-Bertlvise, "Android-based Simulator to Support Tomasulo Algorithm Teaching and Learning", International Journal of Computer Applications (IJCA), Vol. 170, No. 2, pp. 24-29, doi>10.5120/ijca2017914703, July 2017.

[4] Chalk, B. "Evaluation of a Simulator to Support the Teaching of Computer Architecture". 3rd Annual LTSN-ICS Conference, Loughborough University.

[5] Mustafa, B. "Evaluating a System Simulator for Computer Architecture Teaching and Learning Support". ITALICS Vol. 9, issue 1, Feb. 2010, ISSN: 1473-7507.

[6] Grigoriadou M., Kanidis V., Gogoulou A. "A Web-Based Educational Environment for Teaching the Computer Cache Memory", IEEE Transactions on Education, 2006, Vol. 49, No.1, p. 147-156.

[7] V. Luković, R. Krneta, A. Vulović, C. Dimopoulos, K. Katzis, and M. Meletiou-Mavrotheris, "Using Logisim Educational Software In Learning Digital Circuits Design," in Proceedings of 3rd International Conference on Electrical, Electronic and Computing Engineering IcETRAN 2016, p. AUI1.5.1-6.

[8] "Tomasulo's Algorithm. University of Edinburgh". Available at: http://www.icsa.inf.ed.ac.uk/research/groups/hase/models/tomasulo/index.html. Accessed on Feb. 2017.

[9] Typanski N., "Tomasulo algorithm simulator (prototype)". Available at: http://nathantypanski.github.io/tomasulo-simulator/ Accessed on Feb. 2017.

[10] University of Massachusetts at Amherst. "Dynamic Scheduling Using Tomasulo's Algorithm". Available at: http://www.ecs.umass.edu/ece/koren/architecture/. Accessed on Feb. 2017.

[11] "Tomasulo's Algorithm for Dynamic Scheduling". Available at: http://dark.eit.lth.se/darklab/tomasulo/script/tomasulo.htm. Accessed on Feb. 2017.

[12] Gomez-Luna, J., Herruzo, E. and Benavides, J. I., 2009. MESI Cache Coherence Simulator for Teaching Purposes. CLEI ELECTRONIC JOURNAL. 12, 1.

[13] Laguens, A. A., Mir, S.B. and Quintana Orti, E.S., 2011. An Interactive Animation for Learning How Cache Coherence Protocols Work. In proceedings of INTED2011 Conference, 7-9 March, Valencia Spain.