A Survey and Evaluation of Simulators Suitable for Teaching Courses in Computer Architecture and Organization

Boško Nikolic, *Member, IEEE*, Zaharije Radivojevic, Jovan Djordjevic, and Veljko Milutinovic, *Fellow, IEEE*

***Abstract*—Courses in Computer Architecture and Organization are regularly included in Computer Engineering curricula. These courses are usually organized in such a way that students obtain not only a purely theoretical experience, but also a practical understanding of the topics lectured. This practical work is usually done in a laboratory using simulators of computer systems. Since the open literature contains a variety of simulators being used for such purposes, this paper attempts to give a survey of simulators suitable for teaching courses in computer architecture and organization, to establish the evaluation criteria and to evaluate selected simulators according to these criteria.**

***Index Terms*—Computer architecture, computer science education, computer-aided instruction, educational technology, simulation.**

I. Introduction

COMPUTER architecture and organization is one of the key knowledge areas in Computer Engineering. Therefore, courses in this area have to achieve multiple objectives. The basic objective is that these courses must provide an overview of computer architecture and organization concepts and give insight into the operation of a typical computer system. Beyond this, they should highlight all the important issues in computer architecture and organization that practicing engineers are faced with. The courses are also expected to reinforce topics that are common to other areas of computer science, such as programming languages, operating systems, data base systems, and so on. In addition they must acquaint students with the various tools they need to be familiar with, in order to be able to carry out research and development once they leave school.

Consequently, most courses in computer architecture and organization include practical work in the laboratory. This work makes it possible for students to verify their theoretical knowledge from lecture classes, by observing and exploring characteristics and behaviors of actual systems. The practical work also includes designing, implementing, testing, and documenting hardware and software, designing experiments to acquire data, analyzing and interpreting that data, and in some cases using that data to correct or improve the design. Introductory laboratories are somewhat directed to reinforce concepts presented in lecture classes. Therefore, activities in the laboratory are mainly directed to demonstrate specific phenomena or behavior, and provide experiences with measuring and studying desired characteristics. However, intermediate and advanced laboratories include problems that are more open-ended. As a result, students are requested to design and implement solutions, to design experiments to acquire data needed to complete the design or to measure various characteristics. All these activities in the laboratory are most effectively done using appropriate simulators of computer systems.

The organization of a course requires decisions to be made with respect to the selection of topics to be covered, the allocation of hours per topic, the distribution of hours per lecture, practical exercises, laboratory work, the selection of course texts, the creation of assignments, and so on. In the case where laboratory work is part of a course, the selection of a computer simulator is a very important issue. The choice is between using an available simulator or developing a new one. In both cases some acquaintance with the basic features of available simulators is required. As an aid in making such decisions, this paper attempts to give a survey of simulators suitable for teaching courses in computer architecture and organization, to establish the evaluation criteria and to evaluate selected simulators according to the criteria. The information used to do this is based solely on material presented in the literature; the simulators were not otherwise obtained, evaluated and compared by the authors.

The rest of the paper is organized as follows. Section II gives a survey of simulators suitable for teaching computer architecture and organization. Section III introduces the criteria for their evaluation, that fall into two groups: the topics covered and the simulation features. Section IV presents the evaluation results for the simulators surveyed. Section V concludes the paper.

II. Survey of Selected Simulators

The open literature offers a variety of simulators suitable for teaching courses in computer architecture and organization. The basic characteristics of selected simulators are shown in Table I and include authors, target users, operating systems on which the simulators run, programming languages used for their development, availability and target computer architecture. Their detailed descriptions are not given in the paper for reasons of length. In order to provide readers with the complete features of these simulators, more data are given in [1].

Manuscript received January 17, 2008; revised August 13, 2008. First published May 26, 2009; current version published November 04, 2009.

The authors are with the School of Electrical Engineering—Computer Engineering and Information Theory, Belgrade 11000, Serbia (e-mail: bosko.nikolic@etf.bg.ac.yu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TE.2008.930097

TABLE I
CHARACTERISTICS OF SELECTED SIMULATORS

System	Author	Users	Operating System	Programming language	Availability	Target Computer Architecture
ANT	Harvard University, USA	Students	Windows, Linux, and MacOS	C, Java	Free AC [2]	Custom MIPS R2000 like architecture
CASLE	Purdue University, USA	Students	Windows, Linux, and MacOS	Java	NA [3]	Custom architecture
CCSTUDIO	Texas Instruments Inc.	Hardware and software developers, students, and testers	Windows Vista/XP/2000	NA	Commercial, free trial [4]	ARM, C2000, C5000, C6000, and OMAP
CodeWarrior	Freescale Semiconductor, Inc.	Hardware and software developers, students, and testers	Windows, Linux, Solaris, and MacOS	NA	Commercial, free trial [5]	HC(S)08/RS08, 56800/E, 5xx/55xx, 68K, ColdFire, OSEK, Power Architecture, PPC, StarCore and SDMA
CPU Sim	Colby College, USA	Students	Windows, Linux, and MacOS	Java	Free [6]	RISC-like accumulator-based architectures
DigLC2	University Paris-Sud, France	Students	Windows and UNIX	C	GPLv2 [7]	Custom LC-2 architecture
DLXview	Purdue University, USA	Students	Solaris, SunOS, HP-UX, and Linux	C	GPLv2 [8]	DLX architecture
Easy CPU	Holon Institute of Technology, Israel	Students and high school pupils	Windows	Java	NA [9]	Simplified Intel 80X86 architecture
EDCOMP	University of Belgrade, Serbia	Students	Windows, Linux, and MacOS	Java	Free [10]	Custom CISC architecture
ESCAPE	Ghent University, Belgium	Students	Windows	Delphi	Free [11]	Custom von Neumann architecture and Harvard architecture
FastCache	University of Wisconsin Madison, USA	Students	Solaris 2.5.1 or older	C	Free [12]	-
HASE	University of Edinburgh, UK	Students, developers, testers, educators, and architecture designers	Solaris, Linux, and Windows	Java	Free AC [13]	-
HASE-Dinero	University of Edinburgh, UK	Students	Solaris, Linux, and Windows	HASE++	Free [14]	-
ISE Design Suite	Xilinx Inc.	Hardware and software developers, students, testers, and performance analytics	Windows, Red Hat Enterprise Linux, and SUSE Enterprise Linux	NA	Commercial, free for 60-day [15]	-
JCachesim	University of Siena, Italy	Students	Windows, Linux, and MacOS	Java	Free [16]	MIPS R3000
JHDL	Brigham Young University, USA	Students, developers, testers, and those involved in performance analysis	Windows, Linux, and MacOS	Java	OSSLA [17]	-
Logisim	Hendrix College,	Students	Windows, Linux,	Java	GPL [18]	-
	USA		and MacOS			
M5	The University of Michigan, USA	Hardware and software developers, students, testers, and performance analytics	Linux, MacOS X, Solaris, OpenBSD, and Cygwin	Python, C++	BSD [19]	Alpha, SPARC, MIPS, and ARM ISAs
Quartus II	Altera Corporation	Hardware and software developers, students, testers, and performance analytics	Windows, Red Hat Enterprise Linux, SUSE Enterprise Linux, and HP-UX	NA	Commercial, 30-day free trial [20]	-
RM	University of Catalonia (UPC), Spain	Students	Windows and Linux	NA	Free [21]	Custom RISC architecture
RSIM	Rice University Houston Texas, USA	Students and researchers	UNIX compatible platforms	C, C++	UIUC/NCSA [22]	Custom architecture
SIMCA	University of Minnesota, USA	Students, developers, and testers	SUN SunOS 5.6 and IRIX 6.2	C	Free [23]	Custom superthreaded architecture
SimFlex	Carnegie Mellon University, USA	Software developers, students, testers, and performance analytics	Linux	C++	Free [24]	Architecture supported by Simics
Simics	Virtutech AB Stockholm, Sweden	Software developers, students, testers, performance analytics	Linux (x86, PowerPC, and Alpha), Solaris/UltraSparc, True64/Alpha, and Windows 2000/x86	Python, C	Commercial, Free AC [25]	PowerPC, x86, ARM, and MIPS
SimOS	Stanford University, USA	Students, developers, performance analytics	Irix, Solaris, and Linux	C	Free NC [26]	MIPS-based multiprocessors architecture
SimpleScalar	University of Wisconsin-Madison, USA	Students, developers, and performance analytics	UNIX and Windows	C	Free NC [27]	Alpha, PISA, ARM, and x86
SMOK	University of Washington, USA	Students	Windows	NA	Free [28]	-
Virtual Vulcan	YOERIC Corporation	Students	Windows	NA	Commercial [29]	-

Legend: NA – Not Available, **Free AC** – Free for Academic Institutions, **OSSLA** – Open Source Software License Agreement, **GPL** – GNU General Public License, **GPLv2** – GNU General Public License, Version 2, **BSD** – Berkeley-Style Open Source License, **UIUC/NCSA** – University of Illinois/NCSA Open Source License, **Free NC** - Free For Non-Commercial Academic Use

Generally, these simulators can be separated into two major groups.

The first group contains the appropriate tools and necessary methods to enable the user first to build specific computer

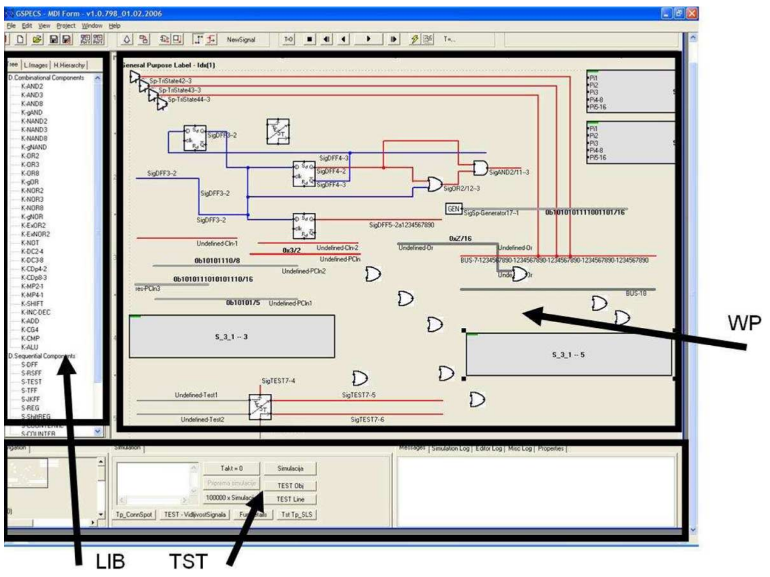


Fig. 1. The main window of a first group simulator.

system configurations and then to simulate them. Although some simulators were designed for general-purpose digital logic circuit design, while others were developed explicitly for modeling and studying computer architecture and organization, it was decided to treat them as a single group. A simulator currently being developed by the authors is taken as an illustration of features usually offered by these simulators [30]. The main window of the simulator (Fig. 1) is separated into three areas: LIB—Available libraries (top left), WP—Working panel (top right), and TST—Testing options (bottom).

This type of simulator implies five-step procedure in which a user:

1. selects components from the library;
2. places selected components onto the working panel;
3. connects the selected component with other ones in order to create a new component;
4. creates an interface for the created new component; and
5. tests the created component.

A component created in this manner will appear in the LIB area and can be used later as a building component.

The most representative simulators from this group are: HASE, ISE Design Suite, JHDL, Logisim, M5, Quartus II, Simics, SMOK, and Virtual Vulcan. Hierarchical Computer Architecture design and Simulation Environment (**HASE**) is a set of tools for creating hierarchical modules, configuring them, simulating obtained systems and analyzing a scalable architecture [13], [31]. **ISE Design Suite** enables the logic design with the timing closure for higher performance and lower power designs, embedded systems design using wizards and DSP design [15]. **JHDL** (structurally based Hardware Description Language implemented with Java) is a set of FPGA CAD tools that allows the user to design the structure and layout of a circuit, to debug the circuit and to perform the bit-stream synthesis [17], [32], [33]. **Logisim** is an educational tool for designing and simulating digital logic circuits [18], [34], [35]. **M5** provides features necessary for deterministically simulating networked hosts, including the full-system capability and a detailed I/O subsystem, and multiple networked systems [19], [36]. **Quartus II** is an environment for the design of FPGAs, CPLDs and structured ASICs [20]. **Simics** is a platform for a full system simulation that can run the actual firmware and the completely unmodified kernel and driver code [25], [37]. SLOOP Machine Organization Kit (**SMOK**) is a toolkit that provides the interface for constructing and debugging machine models [28], [38]. **Virtual Vulcan** is a software simulation of an expandable breadboard with support for more than 75 digital circuits [29], [39].

The second group of simulators contains appropriate tools that enable the user to simulate already created systems. A simulator developed by the authors is taken as an illustration of features usually offered by these simulators [10], [40].

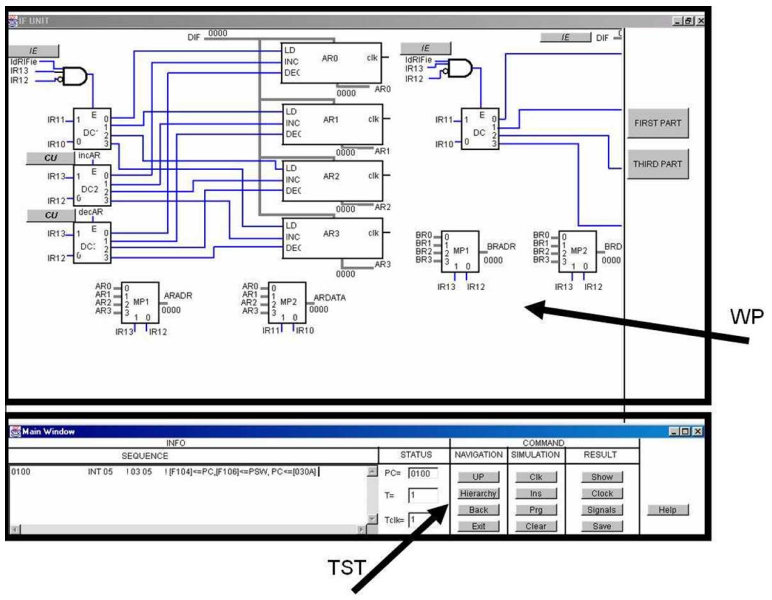
The main window of the simulator (Fig. 2) is separated into two areas: WP—Working panel (top) and TST—Testing options (bottom).

This type of simulator implies the five step procedure in which a user:

1. initializes the simulator with default options;
2. specifies the test configuration;
3. creates test vectors and diagrams to be observed;
4. carries out the simulation; and
5. analyzes test results for the specified test configuration.

This type of simulator usually has some configurable parameters that have to be set before the simulation is carried out. These parameters are usually the number of peripherals, the initial values of registers and memory locations, and so on; they make it possible to specify a variety of computer configurations for testing. Test reports are typically given as tables containing register/memory contents, timing diagrams, textual files with various traces, etc.

The most representative simulators from this group used in the evaluation are: ANT, CASLE, CCSTUDIO, CodeWarrior, CPU Sim, DigLC2, DLXview, Easy CPU, EDCOMP, ESCAPE, FastCache, HASE-Dinero, JCachesim, RM, RSIM, SIMCA, SimFlex, SimOS, and SimpleScalar. **ANT** is a virtual



```
0100            INT 05     ! 03 05     ! [F104]<=PC,[F106]<=PSW, PC<=[030A]
```

PC= 0100
T= 1
Tclk= 1

Fig. 2. The main window of a second group simulator.

machine, based on a basic RISC architecture [2], [41]. Compiler/Architecture Simulation for Learning and Experimenting Simulator (**CASLE**) is a machine language level simulator [3]. Code Composer Studio (**CCSTUDIO**) is an integrated development environment for developing DSP code for the TMS320 DSP, ARM, and OMAP processor family [4], [42]. **CodeWarrior Development Studio** with Integrated Development Environment provides tools for creating, compiling, linking, assembling, and debugging the 8- and 16-bit family of microcontrollers with Instruction Set Simulator [5]. **CPU Sim** is a computer simulation package that allows the user to specify the processor details [6], [43]. **DigLC2** is a gate-level simulator, which provides a detailed description of all processor components [7], [44]. **DLXview** is a pipeline simulator that uses the DLX set of instructions for three versions of the DLX pipeline [8], [45]. **Easy CPU** is a simplified instruction set Intel 80×86 simulator with an animated presentation of the internal computer operations [9], [46], [47]. EDucation COMPuter (**EDCOMP**) is a flexible computer system with the RTL level of simulation [10], [40]. Environment for the Simulation of Computer Architectures for the Purpose of Education (**ESCAPE**) is an environment that supports the simulation of two custom-made processor architectures [11], [48]. **FastCache** provides a framework for implementing memory system simulators [12], [49]. **HASE Dinero** presents an environment for cache memory simulation [14], [50], [51]. **JCachesim** is a simulation environment for the performance evaluation of the MIPS based computer systems with a cache memory [16], [52]. **RM** (Rudimentary Machine) simulates a basic RISC style system [21], [53], [54]. Rice Simulator for ILP Multiprocessor (**RSIM**) is an event driven simulator for the analysis of shared memory multiprocessor and single processor systems with instruction level parallelisms [22], [55], [56]. The Simulator for Multi-threaded Computer Architecture (**SIMCA**) supports the performance evaluation of super-threaded architectures and the exploration of different design alternatives [23], [57], [58]. **Sim-Flex** is a simulation framework that uses a component-based design [24], [59]–[61]. **SimOS** is a complete computer system simulation environment designed for the efficient and accurate study of both single-processor and multiprocessor computer systems [26], [62]. **SimpleScalar** is a toolset with a compiler, linker, simulator, and visualization tools [27], [63], [64].

III. Evaluation Criteria

The evaluation of simulators requires the use of relevant criteria. In the absence of any known criteria used for such purposes, the authors have established criteria which fall into two groups: the coverage of topics and the simulation features. The coverage criteria evaluate the topics studied in computer architecture and organization courses that a simulator supports, while the simulation features criteria analyze the functionality of selected simulators. Other criteria were also considered, such as the learning curve and the ease of use of the simulator, student performance metrics in courses that have used the simulator, and the like. However, the inclusion of these criteria requires actual use and evaluation of the simulators on course projects, which is impractical. Therefore, the criteria selected for this survey are limited to those that could be evaluated solely from information available in the open literature.

The topics coverage criteria are established using the IEEE Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering [65], which presents curriculum guidelines defining broad knowledge areas that are applicable to all computer engineering programs. Computer Architecture and Organization is one of these knowledge areas, which comprises a set of 10 knowledge units. Within each knowledge unit, a set of topics and appropriate core hours are given. The six knowledge units with five or more core hours are: *Fundamentals of computer architecture*, *Memory system organization and architecture*, *Interfacing and communication*, *Device subsystems*, *Processor systems design*, and *Organization of the CPU*. These

TABLE II
LIST OF TOPICS IN EACH KNOWLEDGE UNIT

Fundamentals of computer architecture	Organization of the von Neumann machine Instruction formats The fetch/execute cycle; instruction decoding and execution Registers and register files Instruction types and addressing modes Subroutine call and return mechanisms Programming in assembly language I/O techniques and interrupts Other design issues
Memory system organization and architecture	Memory systems hierarchy Coding, data compression, and data integrity Electronic, magnetic and optical technologies Main memory organization and its characteristics and performance Latency, cycle time, bandwidth, and interleaving Cache memories (address mapping, line size, replacement and write-back policies) Virtual memory systems Memory technologies such as DRAM, EPROM, FLASH Reliability of memory systems; error detecting and error correcting systems
Interfacing and Communication	I/O fundamentals: handshaking, buffering, Techniques: programmed & interrupt-driven I/O, DMA Interrupt structures: vectored and prioritized, interrupt overhead, interrupts and reentrant code Memory system design and interfacing Buses: bus protocols, local and geographic arbitration
Device subsystems	External storage systems; organization and structure of disk drives and optical memory Basic I/O controllers such as a keyboard and a mouse RAID architectures Video control I/O Performance SMART technology and fault detection Processor to network interfaces
Processor systems design	The CPU interface: clock, control, data and address buses Address decoding and memory interfacing Basic parallel and serial interfaces Timers System firmware
Organization of the CPU	Implementation of the von Neumann machine Single vs. multiple bus datapaths Instruction set architecture; machine architecture as a framework for encapsulating design decisions Relationship between the architecture and the compiler Implementing instructions Control unit: hardwired realization vs. microprogrammed realization Arithmetic units, for multiplication and division Instruction pipelining Trends in computer architecture: CISC, RISC, VLIW Introduction to instruction-level parallelism (ILP) Pipeline hazards: structural, data and control Reducing the effects of hazards

knowledge units are chosen as the topics coverage criteria and given with detailed lists of their topics in Table II.

The simulation features criteria are established on the basis of the most commonly found features in the selected simulators.

Some of the selected simulators have been developed for introductory courses in Computer Architecture and Organization and cover basic concepts. On the other hand, there are simulators devised for courses in the final years of graduate studies and/or postgraduate studies, which deal with advanced topics. As a result, simulators differ greatly in scope and complexity. The criterion adopted to reflect this is titled *Scope and Complexity*. The levels of assessment are Basic Architecture (BA) and Advanced Architecture (AA).

One group of simulators includes tools supporting the design of reusable computer system modules and their simulation. Other simulators facilitate only the simulation of predefined systems. Consequently, simulators could be divided into those simulating either user-defined computer system structures or fixed computer system structures. The criterion adopted here is *Design Support*. The levels of assessment are Yes and No.

Many simulators have a very good visual presentation, suitable to demonstrate the inner working of a computer system. On the other hand, there are simulators without visual presentation where the results are given in textual form with the possibility for their postprocessing. Simulators can therefore be divided into those with or without visual presentation. The cri-

TABLE III
EVALUATION RESULTS FOR THE TOPICS COVERAGE

System	Fundamentals of computer architecture	Memory system organization and architecture	Interfacing and Communication	Device subsystems	Processor systems design	Organization of the CPU	Overall coverage
ANT	81.25	22.22	20.00	0.00	10.00	25.00	28.26
CASLE	62.50	16.67	0.00	0.00	10.00	29.17	22.83
CCSTUDIO	87.50	22.22	30.00	14.29	30.00	20.83	33.70
CodeWarrior	87.50	11.11	30.00	14.29	30.00	16.67	30.43
CPU Sim	87.50	16.67	10.00	0.00	10.00	29.17	28.26
DigLC2	62.50	27.78	30.00	0.00	30.00	29.17	30.43
DLXview	87.50	22.22	0.00	0.00	20.00	58.33	36.96
Easy CPU	81.25	16.67	10.00	0.00	20.00	25.00	27.17
EDCOMP	100.00	33.33	100.00	7.14	40.00	62.50	56.52
ESCAPE	68.75	16.67	10.00	0.00	30.00	33.33	28.26
FastCache	6.25	38.89	0.00	0.00	10.00	0.00	9.78
HASE	31.25	50.00	60.00	0.00	50.00	41.67	38.04
HASE-Dinero	56.25	33.33	20.00	0.00	10.00	25.00	26.09
ISE Design Suite	25.00	66.67	60.00	28.57	60.00	33.33	43.48
JCachesim	37.50	44.44	30.00	0.00	10.00	16.67	23.91
JHDL	25.00	66.67	60.00	28.57	60.00	33.33	43.48
Logisim	18.75	11.11	0.00	0.00	20.00	20.83	13.04
M5	100.00	66.67	90.00	50.00	70.00	54.17	69.57
Quartus II	25.00	61.11	60.00	35.71	60.00	33.33	43.48
RM	75.00	5.56	10.00	0.00	10.00	33.33	25.00
RSIM	68.75	33.33	30.00	0.00	20.00	50.00	36.96
SIMCA	62.50	44.44	10.00	0.00	20.00	41.67	33.70
SimFlex	56.25	33.33	20.00	28.57	30.00	37.50	35.87
Simics	100.00	55.56	100.00	50.00	70.00	37.50	64.13
SimOS	68.75	22.22	50.00	7.14	20.00	33.33	33.70
SimpleScalar	100.00	11.11	10.00	0.00	10.00	54.17	35.87
SMOK	31.25	44.44	10.00	21.43	20.00	25.00	27.17
Virtual Vulcan	18.75	5.56	10.00	14.29	10.00	8.33	10.87

terion adopted is *Visual Presentation*. The levels of assessment are Yes and No.

The implementation of the simulation flow differs among simulators. In some of them the complete simulation session is carried out interactively allowing a user to start and stop the simulation, roll it back, save the simulation context and continue it later, etc. In other ones the complete simulation is performed in batch mode. Simulators can, therefore, be divided into those with interactive or batch simulation flow. The criterion adopted is *Simulation Flow*. The levels of assessment are InteraCtive (IC) and BatCh (BC).

The granularity of a simulation session can be at the clock, instruction or program level. Based on the lowest level of granularity supported during a simulation session, simulators can be divided into those that support simulation on the clock cycle, instruction or program level. The criterion adopted is *Simulation Level*. The levels of assessment are Clock Level (CL), Instruction Level (IL), and Program Level (PL).

All simulators show the structure of the system as a composition of functional blocks. There are simulators where implementation details are not visible and data observed during the simulation remain at the level of information transfer between blocks. However, there are simulators where one can observe the structures of blocks at lower levels of implementation details. Based on the level of details visible during the simulation, simulators can be divided into those with or without visible implementation details of the computer system. The criterion adopted is *Implementation Details*. The levels of assessment are Yes and No.

An Internet interface and support for distance learning have become a standard issue in modern university courses. Consequently, the possibility of using a simulator via Internet be-

TABLE IV
EVALUATION RESULTS FOR THE SIMULATION FEATURES

System	Scope and Complexity	Design Support	Visual Presentation	Simulation Flow	Simulation Level	Implementation Details	Distance Learning
ANT	AA	No	No	BC	IL	No	No
CASLE	BA	No	No	BC	IL	No	Yes
CCSTUDIO	**AA**	**No**	**No**	**IC**	**IL**	**No**	**No**
CodeWarrior	**AA**	**No**	**No**	**IC**	**IL**	**No**	**No**
CPU Sim	BA	No	No	IC	CL	Yes	No
DigLC2	BA	No	Yes	IC	IL	Yes	No
DLXview	AA	No	Yes	IC	CL	Yes	No
Easy CPU	BA	No	Yes	IC	CL	No	Yes
EDCOMP	AA	No	Yes	IC	CL	Yes	Yes
ESCAPE	AA	No	Yes	IC	CL	Yes	No
FastCache	BA	No	No	BC	PL	No	No
HASE	AA	Yes	Yes	IC	CL	No	Yes
HASE-Dinero	AA	No	Yes	IC	CL	No	Yes
ISE Design Suite	AA	Yes	Yes	IC	CL	Yes	No
JCachesim	BA	No	Yes	BC	IL	No	Yes
JHDL	AA	Yes	Yes	IC	CL	Yes	No
Logisim	BA	Yes	Yes	IC	CL	No	No
M5	AA	Yes	No	BC	CL	Yes	No
Quartus II	AA	Yes	Yes	IC	CL	Yes	No
RM	BA	No	Yes	IC	CL	Yes	No
RSIM	AA	No	No	BC	CL	No	No
SIMCA	AA	No	No	BC	PL	No	No
SimFlex	AA	No	No	BC	IL	No	No
Simics	AA	Yes	No	BC	IL	No	No
SimOS	AA	No	No	BC	PL	No	No
SimpleScalar	AA	No	No	BC	CL	No	No
SMOK	AA	Yes	Yes	IC	CL	Yes	No
Virtual Vulcan	BA	Yes	Yes	IC	CL	No	No

comes an important criterion for the simulator evaluation. Based on the availability of support for distance learning, simulators can be divided into those with and without support for distance learning. The criterion adopted is *Distance Learning*. The levels of assessment are Yes and No.

Consequently, the desirable simulation features would offer a simulator that supports the advanced architecture of computer systems and custom computer system design, provides the visual presentation and interactive flow, performs the simulation on the program, instruction, and clock cycle level and offers implementation details and distance learning facilities.

IV. EVALUATION OF SELECTED SIMULATORS

Evaluation of selected simulators is carried out separately using the topics coverage criteria and the simulation features criteria. The evaluation results are summarized in Tables III and IV, while more details concerning the topics coverage and the methodology used in obtaining percentages are given in [1]. The results of the evaluation are discussed briefly here.

Topics in the Fundamentals of Computer Architecture unit are covered to a great extent in almost all simulators (Table III). Some of the simulators, such as EDCOMP, M5, Simics, and SimpleScalar, have 100% coverage of the recommended topics in this knowledge unit. The exceptions are simulators such as FastCache (6.25%), Logisim (18.75%), and Virtual Vulcan (18. 75%), which are developed for specialized needs and are not developed to deal with the topics from this unit.

The memory system organization and architecture unit includes diversified topics, which is probably why none of the simulators covers all topics. The best coverage is achieved with simulators such as ISE Design Suite(66.67%), JHDL (66.67%), M5 (66.67%), Quartus II (61.11%), and Simics (55.56%), which, however, do not deal with topics of different memory technologies, and reliability and error correction.

Topics in the Interfacing and Communication unit, the Device subsystems unit and the Processor systems design are generally poorly supported by simulators. A probable explanation is that most of the simulators analyzed concentrate mainly on topics related to the basic components of a computer system that include processor, main memory, and simple input/output devices. The exceptions are those simulators that are specifically developed to deal with some of the topics in these three units. As a result, the Interfacing and Communication unit has a very good coverage with simulators EDCOMP (100.00%), Simics (100.00%), and M5 (90.00%), the Device subsystems unit with simulators M5 (50.00%) and Simics (50.00%) and the Processor systems design unit with simulators M5 (70.00%), Simics (70.00%), ISE Design Suite (60.00%), JHDL (60.00%), and Quartus II (60.00%).

Topics in the Organization of the CPU unit come very near to the average coverage of topics in the Fundamentals of Computer Architecture unit. This result probably stems from the fact that these two units are complementary and, as stated in the previous paragraph, deal with the basic components of a computer system, which are covered by most of the simulators. The best coverage is achieved with simulators EDCOMP (62.50%), DLXview (58.33%), M5 (54.17%), SimpleScalar (54.17%), and RSIM (50.00%).

The evaluation results given in Table III can be looked at in two ways: the overall topics coverage and the coverage per unit. The best overall topics coverage is achieved with simulators M5 (69.57%), Simics (64.13%), EDCOMP (56.52%), ISE Design Suite (43.48%), JHDL (43.48%), and Quartus II (43.48%). However, as already discussed, there are simulators with lower overall topics coverage, but with high coverage for a specific unit. This kind of information is more important than the overall coverage for courses limited to a specific unit.

Simulators with basic architecture, such as CASLE, CPU Sim, DigLC2, Easy CPU, and RM are developed with the primary objective of enabling a user to get acquainted with the functioning of a computer system (Table IV). The simulators mainly cover the operations of the processor, and show how the instruction fetch and decode, operand address calculation and operand fetch, and the operation execution phases are performed. Simulators FastCache, JCachesim, and Logisim deal with specific topics, such as cache memory (FastCache and JCachesim) and switching circuits (Logisim). Some simulators with advance architecture support (HASE, ISE Design Suite, JHDL, M5, Quartus II, Simics, and Virtual Vulcan) enable a user to define the complexity of the system, which can range from simple to highly complex. Others (ANT, DLXview, EDCOMP, HASE-Dinero, RSIM) simulate concrete systems of great complexity. There are also simulators (CCSTUDIO, CodeWarrior, SIMCA, Simics, SimOS, SimpleScalar) which simulate existing hardware platforms.

Simulators with design support differ in the initial set of available components and procedures for their use for creating new components. The initial set of components in the case of Logisim and SMOK includes basic gates like AND, OR, NOT, D FF, and Tri-State, while HASE, ISE Design Suite, Quartus II, and JHDL include complex elements like comparators, counters, memory chips, etc. In addition, these simulators make it possible to use components from user-defined files written in some hardware description language. On the other hand, Virtual Vulcan offers a selection of commercially available chips like 7400 Q 2 NAND, 7404 Hex Inverter, 7451 AND-OR-Invert, 7474 Dual D FF, 74147 10 to 4 line BCD, 74399 Q 2-Port Reg, etc.

Visual presentation, simulation flow and simulation level are interrelated criteria. A considerable number of simulators with visual presentation (DLXview, Easy CPU, EDCOMP, ESCAPE, HASE, HASE-Dinero, ISE Design Suite, JHDL, Logisim, Quartus II, RM, SMOK, Virtual Vulcan) enable interactive flow and clock level simulation. DigLC2 and JCachesim carry out simulation at the instruction level, because their primary aim is not the presentation of implementation details. Simulators with only textual interface and batch simulation flow carry out the simulation at the program level (FastCache, SIMCA, SimOS), the instruction level (ANT, CASLE, CCSTUDIO, CodeWarrior, M5, SimFlex, Simics) and the clock level (RSIM, SimpleScalar). These simulators are more frequently used for the performance analysis of simulated systems.

Implementation details are available in eleven simulators. Six of them (CPU Sim, DigLC2, DLXview, EDCOMP, ESCAPE, RM) do not have any design support, while the remaining five (ISE Design Suite, JHDL, M5, Quartus II, SMOK) do. CPU Sim, DigLC2 and RM present implementation details of RISC based processors, DLXview and ESCAPE of processors with pipeline organization, and EDCOMP of CISC based processors. The realization of such a simulator generally requires a significant effort both to design a particular part of a computer system and to develop its simulator. Therefore, this feature is usually implemented only in those simulators where the targeted users really need it.

Distance learning is a significant trend in modern university courses. As a result, a few simulators (CASLE, Easy CPU, EDCOMP, HASE, JCachesim) have been developed as WEB-based applications. These simulators have been implemented using the Java programming language, which offers various techniques suitable for Internet access. In line with this trend, some of earlier developed simulators (HASE, CASLE) have been rewritten in Java. With the development of new programming frameworks, porting existing simulators, and the development of new simulators with Internet accessibility, will increase.

The evaluation results given in Table IV show that generally each of the simulators offers a good set of simulation features. EDCOMP, HASE, ISE Design Suite, JHDL, M5, and Quartus II meet most of the criteria. This level of capability is probably the result of recent developments in software technologies, which include advanced visual builder tools, object oriented methodologies, efficient concurrent programming, WEB programming, etc.

V. CONCLUSION

Computer architecture and organization is one of the key knowledge areas in Computer Engineering. Most of courses in computer architecture and organization include not only lecture classes but also practical work using simulators of computer systems. This paper surveyed and evaluated a range of such simulators. A survey was made of simulators available in the open literature and suitable for laboratory use. Two groups of criteria, topics coverage and the simulation features, were established. Finally, these simulators were evaluated.

The evaluation, based on the topics coverage criteria, shows that there is no single simulator which covers all topics. The best overall topics coverage is achieved with simulators M5 (69.57%) and Simics (64.13%), while most of other simulators achieve about 30.00% coverage. This result is probably the consequence of the fact that the area of computer architecture and organization includes a large number of diversified topics and that the development of a simulator that covers all topics would result in a system that is very large and impractical to use.

The evaluation, based on the simulation features criteria, shows that there are many simulators which meet a great number of these criteria. Simulators EDCOMP, HASE, ISE Design Suite, JHDL, M5, and Quartus II meet most of the criteria. This wide range of effective simulators probably stems from the fact that developers nowadays have many modern technologies available for the realization of complex, visual, and WEB-based systems that can be used for the development of simulators.

REFERENCES

[1] B. Nikolic, Z. Radivojevic, J. Djordjevic, and V. Milutinovic, Survey of Simulators 2008 [Online]. Available: http://rti.etf.bg.ac.yu/rti/ri3aor/Simulators/index.html
[2] Welcome to the Ant Home Page Harvard Univ., Cambridge, MA, 2005 [Online]. Available: http://www.ant.harvard.edu/
[3] G. B. Adams, III, Casle 2004 [Online]. Available: http://cobweb.ecn.purdue.edu/~gba/Index.html
[4] Code Composer Studio User's Guide Texas Instrum., 2008 [Online]. Available: http://focus.ti.com/lit/ug/spru328b/spru328b.pdf
[5] CodeWarrior Development Tools Freescale Semiconductor, 2008 [Online]. Available: www.freescale.com/codewarrior
[6] D. Skrien, CPU Sim Home Page 2008 [Online]. Available: http://www.cs.colby.edu/djskrien/CPUSim/
[7] A. Cohen and O. Temam, DigLC2—Gate-Level LC-2 Simulator 2005 [Online]. Available: http://www-rocq.inria.fr/~acohen/teach/diglc2.html.en
[8] G. Adamas, DLXview v0.9—Home Page 2005 [Online]. Available: http://yara.ecn.purdue.edu/~teamaaa/dlxview/
[9] Easy CPU H.I.T.- Holon Inst. Technol., 2008 [Online]. Available: http://www.hit.ac.il/EasyCPU/

[10] J. Djordjevic, B. Nikolic, and A. Milenkovic, Computer Architecture Lab. 2006 [Online]. Available: http://rti.etf.bg.ac.yu/rti/ef2ar/labvezbe/CISCA.zip

[11] P. Verplaetse and J. V. Campenhout, ESCAPE v1.1 Homepage 2008 [Online]. Available: http://trappist.elis.ugent.be/~hvdieren/escape/

[12] A. R. Lebeck and D. A. Wood, FastCache 2005 [Online]. Available: http://www.cs.duke.edu/~alvy/fast-cache/

[13] HASE—A Computer Architecture Simulation Environment Inst. Comput. Syst. Architect., School of Inform., Univ. Edinburgh, Edinburgh, Scotland, 2005 [Online]. Available: http://www.icsa.inf.ed.ac.uk/research/groups/hase/

[14] COMPUTER ARCHITECTURE: HASE Dinero Inst. Comput. Syst. Architect., School of Inform., Univ. Edinburgh, Edinburgh, Scotland, 2005 [Online]. Available: http://www.icsa.informatics.ed.ac.uk/research/groups/hase/projects/dinero/

[15] ISE Design Suite 10.1 Xilinx Inc., 2008 [Online]. Available: http://www.xilinx.com/products/design_resources/design_tool/index.htm

[16] R. Giorgi, JCacheSim-Univ. Siena, Faculty of Comp. Eng. 2005 [Online]. Available: http://www.dii.unisi.it/~giorgi/jcachesim/

[17] JHDL Overview Brigham Young Univ., Salt Lake City, UT, 2005 [Online]. Available: http://www.jhdl.org/overview.html

[18] C. Burch, Logisim 2007 [Online]. Available: www.cburch.com/logisim/

[19] N. L. Binkert, R. G. Dreslinski, L. R. Hsu, K. T. Lim, A. G. Saidi, and S. K. Reinhardt, Main Page -M5 2007 [Online]. Available: http://www.m5sim.org

[20] Quartus II Software Altera Corp., 2008 [Online]. Available: http://www.altera.com/products/software/products/quartus2/qts-index.html

[21] Página Oficial de La Máquina Rudimentaria Univ. Politècnica de Catalunya, 2008 [Online]. Available: http://docencia.ac.upc.edu/eines/MR/

[22] S. Adve, RSIM Home Page 2008 [Online]. Available: http://rsim.cs.uiuc.edu/rsim/

[23] J. Huang, SIMCA Home Page 2008 [Online]. Available: http://agassiz.cs.umn.edu/Tools/SIMCA/simca.html

[24] SimFlex: Fast and Accurate Scalable Simulation Elect. Comput. Eng. Carnegie Mellon Univ., Pittsburgh, PA, 2007 [Online]. Available: http://www.ece.cmu.edu/~simflex/

[25] Virtutech Simics—Embedded Systems Simulation Platform, Virtual Hardware for Embedded Software Development Virtutech, 2005 [Online]. Available: http://www.virtutech.com/

[26] R. Bosch, The SimOS Home Page 2005 [Online]. Available: http://simos.stanford.edu/

[27] SimpleScalar LLC The SimpleScalar Toolset, 2005 [Online]. Available: http://www.simplescalar.com

[28] B. Dugan and J. Zahojan, SMOK/Cebollita Home Page 2005 [Online]. Available: http://www.cs.washington.edu/homes/zahor)

[29] YOERIC Corporation—WinBreadboard YOERIC Corp., 2005 [Online]. Available: http://www.yoeric.com/virtualvulcan.htm

[30] N. Grbanovic, J. Djordjevic, and B. Nikolic, Logic Design Lab. 2008 [Online]. Available: http://rti.etf.bg.ac.yu/rti/oo1pot/simulator/IGoV-SoDS_SVE_v1.1.228_17.02.2008.zip

[31] R. N. Ibbett, "HASE DLX simulation model," *IEEE Comput. Soc., IEEE Micro, Special Issue on Comput. Architect. Educ.*, vol. 20, no. 3, pp. 38–47, May/Jun. 2000.

[32] B. Hutchings, P. Bellows, J. Hawkins, S. Hemmert, B. Nelson, and M. Rytting, "A CAD suite for high-performance FPGA design," in *Proc. 7th Ann. IEEE Symp. Field-Programm. Custom Comput. Mach.*, 1999.

[33] P. Bellows and B. Hutchings, "JHDL—An HDL for reconfigurable systems," in *Brigham Young Univ., Symp. FPGAs for Custom Computing Mach.*, 1998, pp. 175–184.

[34] C. Burch, "Logisim: A graphical system for logic circuit design and simulation," *J. Educat. Resources in Comput.*, vol. 2, no. 1, pp. 5–16, 2002.

[35] C. Burch and L. Ziegler, "Science of computing suite (SOCS): resources for a breadth-first introduction," in *Proc. ACM SIGCSE Bull.*, 2004, pp. 437–441.

[36] N. L. Binkert, R. G. Dreslinski, L. R. Hsu, K. T. Lim, A. G. Saidi, and S. K. Reinhardt, "The M5 simulator: Modeling networked systems," *IEEE Micro*, vol. 26, no. 4, pp. 52–60, Jul./Aug. 2006.

[37] P. S. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, G. Hållberg, J. Högberg, F. Larsson, A. Moestedt, and B. Werner, "Simics: A full system simulation platform," *Computer*, vol. 35, pp. 50–58, Feb. 2002.

[38] B. Dugan and J. Zahorjan, "The Sloop ISA and the SMOK toolkit," *J. Educ. Resources in Comput. (JERIC)*, vol. 2, no. 1, pp. 49–71, Mar. 2002.

[39] N. Glass, Java Digital Breadboard Simulator: A Simulator for an Educational Electronics Environment 2005 [Online]. Available: http://www.cs.york.ac.uk/netpro/bboard/

[40] J. Djordjevic, B. Nikolic, and A. Milenkovic, "Flexible web-based educational system for teaching computer architecture and organization," *IEEE Trans. Educ.*, vol. 48, pp. 264–273, May 2005.

[41] D. Ellard, D. Holland, N. Murphy, and M. Seltzer, "On the design of a new CPU architecture for pedagogical purposes," in *Proc. Workshop on Comput. Architect. Educ.*, Anchorage, AK, May 2002, pp. 28–34.

[42] Code Composer Studio IDE—CCSTUDIO—TI Tool Folder Texas Instrum., 2008 [Online]. Available: http://focus.ti.com/docs/toolsw/folders/print/ccstudio.html

[43] D. Skrien, "CPU Sim 3.1: A tool for simulating computer architectures for computer organization classes," *ACM J. Educ. Resources in Comput. (JERIC)*, vol. 1, pp. 46–59, Dec. 2001.

[44] A. Cohen and O. Temam, "Digital LC-2: from bits and gates to a little computer," in *Proc. WCAE*, Anchorage, AK, May 2002.

[45] Y. Zhang and G. B. Adams, III, "An interactive, visual simulator for the DLX pipeline," *IEEE Comput. Soc., Tech. Committee on Comput. Architect. Newsl.*, pp. 9–12, Sep. 1997.

[46] C. Yehezkel, W. Yurcik, and M. Pearson, "Teaching computer architecture with a computer-aided learning environment: State-of-the-art simulators," in *Proc. 2001 Int. Conf. Simul. Multimed. Eng. Educ. (ICSEE), Soc. Comput. Simul. (SCS) Press*, Phoenix, AZ, Jan. 2001.

[47] C. Yehezkel, W. Yurcik, M. Pearson, and D. Armstrong, Three Simulator Tools For Teaching Computer Architecture: EasyCPU, Little Man Computer, and RTLSim 2005 [Online]. Available: www.cstc.org/data/resources/195/EasyCPU_LMC_RTLSim.PDF

[48] P. Verplaetse and J. V. Campenhout, "ESCAPE: Environment for the simulation of computer architecture for the purpose of education," *IEEE TCCA Newsl.*, pp. 57–59, July 1999.

[49] A. R. Lebeck and D. A. Wood, "Active memory: A new abstraction for memory system simulation," in *Proc. ACM SIGMETRICS*, May 1995, vol. 7, pp. 220–230.

[50] M. Seymour, The HASE Dinero User Guide 2005 [Online]. Available: http://www.dcs.ed.ac.uk/home/hase/projects/dinero/index.html

[51] R. Ibbett and F. Mallet, "Computer architecture simulation applets for use in teaching," in *Proc. 33rd ASEE/IEEE Frontiers in Educ. Conf.* , Boulder, CO, Nov. 5–8, 2003, p. F1C-20–5 [Online]. Available: fie.engrng.pitt.edu/fie2003/papers/1545.pdf

[52] I. Branovic, R. Giorgi, and A. Prete, "Web-based training on computer architecture: The case for JCachesim," in *Proc. Workshop on Comput. Architect. Educ. (WCAE-98)*, Anchorage, AK, May 2002, pp. 56–60.

[53] E. Pastor, F. Sanchez, and A. M. de Corral, "A rudimentary machine: Experiences in the design of a pedagogic computer," in *Proc. Workshop on Comput. Architect. Educ. (WCAE-98)*, Barcelona, Spain, Jun. 27, 1998.

[54] E. Pastor and F. Sánchez, "La máquina rudimentaria: un procesador pedagógico," *III Jornadas de Enseñanza Universitaria sobre Informática (JENUI'97)*, pp. 395–402, Jun. 1997.

[55] C. Hughes, V Pai, P. Ranganathan, and S. Adve, "RSIM: Simulating shared-memory multiprocessors with ILP processors," *IEEE Computer*, vol. 35, no. 2, pp. 40–49, Feb. 2002.

[56] V. S. Pai, P. Ranganathan, and S. V. Adve, "RSIM: An execution-driven simulator for ILP based shared memory multiprocessors and uniprocessors," *IEEE TCCA Newsl.*, pp. 37–48, Oct. 1997.

[57] J. Huang, D. J. Lilja, and T. Systems, "An efficient strategy for developing a simulator for a novel concurrent multithreaded processor architecture," in *Proc. Six Int. Symp. Model., Anal. Simulat. Comput.* , Montreal, Canada, Jul. 1998, pp. 185–191.

[58] J. Huang, The SImulator for Multithreaded Computer Architecture Release 1.2 Lab. Adv. Res. Comput. Technol. Compilers, Univ. Minnesota, 2000, Technical Report No: ARCTiC-00-05.

[59] T. F. Wenisch, R. E. Wunderlich, M. Ferdman, A. Ailamaki, B. Falsafi, and J. C. Hoe, "SimFlex: Statistical sampling of computer architecture simulation," *IEEE Micro Special Issue on Comput. Architect. Simul.*, vol. 26, no. 4, pp. 18–31, Jul./Aug. 2006.

[60] T. F. Wenisch and R. E. Wunderlich, "SimFlex: Fast, accurate and flexible simulation of computer systems," in *Proc. Int. Symp. Microarchitect. (MICRO-38)*, Barcelona, Spain, Nov. 2005, tutorial no. 1.

[61] N. Hardavellas, S. Somogyi, T. F. Wenisch, R. E. Wunderlich, S. Chen, J. Kim, B. Falsafi, J. C. Hoe, and A. G. Nowatzyk, "SIMFLEX: A fast, accurate, flexible full-system simulation framework for performance evaluation of server architecture," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 31, no. 4, pp. 31–35, 2004.

[62] M. Rosenblum, E. Bugnion, S. Devine, and S. A. Herrod, "Using the SimOS machine simulator to study complex computer systems," *ACM Trans. Model. Comput. Simul.*, vol. 7, no. 1, pp. 78–103, Jan. 1997.
[63] T. M. Austin, "The SimpleScalar toolset as an instructional tool: Experiences and future directions," in *Proc. 4th Ann. Workshop on Comput. Architecture Educ. (WCAE4)*, Las Vegas, NV, Jan. 31, 1998.
[64] D. Burger and T. M. Austin, "The SimpleScalar tool set, version 2.0," *ACM SIGARCH Comput. Architect. News*, vol. 25, pp. 13–25, Jun. 1997.
[65] Computing Curricula—Computer Engineering (CCCE) Task Force IEEE Computer Society/ACM Computing Curriculum—Computer Engineering, 2005 [Online]. Available: http://www.eng.auburn.edu/ece/CCCE/

Bosko Nikolic (M'09) received the B.Sc., M.S., and Ph.D. degrees in electrical engineering from the University of Belgrade, Serbia.

He is currently an Assistant Professor with the Department of Computer Engineering, School of Electrical Engineering, University of Belgrade. His research interests include Internet programming, digital systems simulation, and the programming language Java.

Zaharije Radivojevic received the B.Sc. and M.S. degrees in electrical engineering from the University of Belgrade, Serbia.

He is currently a teaching assistant with the Department of Computer Engineering, School of Electrical Engineering, University of Belgrade. His research interests include computer architecture, digital systems simulation, and grid computing.

Jovan Djordjevic received the B.Sc. degree in electrical engineering from the University of Belgrade, Serbia, and the M.S. and Ph.D. degrees in computer science from the University of Manchester, U.K.

He is currently a Professor of Computer Engineering with the School of Electrical Engineering, University of Belgrade. His research interests include computer architecture, parallel computer systems, digital systems simulation, and distance learning.

Veljko Milutinovic (F'03) received the B.Sc., M.S., and Ph.D. degrees in electrical engineering from the University of Belgrade, Serbia.

He is currently a Professor of Computer Engineering with the School of Electrical Engineering, University of Belgrade. His research interests include VLSI design, computer architecture, semantic web, determining, e-business, and sensor networks.