Building an integrated platform for teaching and learning of Digital logic

M. Čupić
Faculty of Electrical Engineering and Computing, University of Zagreb, Zagreb, Croatia
Marko.Cupic@fer.hr

Abstract – In this paper we describe an integrated platform for teaching and learning of Digital logic. The developed platform is based on web technologies. It helps teachers to prepare learning and assessment materials and to track student progress. It allows students to communicate with teaching staff, to complete laboratory exercises, to learn and to practice using self-assessment interactive problems, all at their own pace and time. We investigated how students perceived several aspects of this platform using anonymous poll. The obtained results are presented and discussed.

I. Introduction

Introduction to digital logic design is important for many curriculums, e.g. computer engineering, computer science, and electrical engineering. But the question of how to teach it effectively is still unanswered. A typical class-and-a-board approach can be significantly augmented due to todays modern technologies. The idea of inclusion of computers and communications technology to enhance the teaching and learning of engineering is a rather old one (e.g. [1]). For a last two decades, a lot of work has been focused in developing a variety of learning materials accessible over web. For example, a set of Web-based tutorials is described in [2], which aims to supplement the spectrum of other learning experiences offered by the typical introductory logic design course.

Digital design classes are increasingly using programmable logic devices (PLD) to augment the educational experience. A recent study described in [3, 4] found that usage of PLD in laboratory had indeed a positive impact on student learning outcomes.

When actual hardware-based laboratories are infeasible (e.g. due to large number of students), simulation based tools can be used as a supplement. According to [5], in some areas simulation is not an effective pedagogical tool because there are lessons to be learnt through reconciling the difference between theory and experiment. However, for teaching digital logic design to undergraduate students, authors in [6] find that simulation is appropriate because of its use in industry. According to [6], for a digital logic class, simulation has significant advantages over practical laboratory work.

To further improve simulation experience, virtual laboratories can be constructed. For example, in [7], a virtual laboratory is described which allows the students to practice on their design, explore in a tutorial fashion various options, and gives them as close as possible a real hands-on experience, through the use of a “virtual board”.

Various other approached have been devised, to further improve the learning outcomes. In [8], the usage of a game based learning is described. Students are required to build a digital circuit based on FPGA technology which controls team's space ship to battle an opposing ship. An interesting attempt to improve the learning outcome by using student's research projects over several years is also described in [9].

Finally, since the digital logic instructors have created a myriad of new, innovative teaching methods. In [10], a standardized assessment tool is described, which tries to provide a means to directly compare students' conceptual learning in different digital logic design courses.

In this paper, we describe a platform for teaching and learning digital logic we use at our university. The platform is composed of several web-based components and is a result of six years of development. Some components are custom-developed just for digital logic education but other are general purpose and can be used for other courses as well. The goal was to create the environment in which the complete course can be handled. For such an environment, it is important to be well suited for teaching staff and for students. So the minimal requirements would be to allow teachers to see enrolled students and to track their progress. Additional requirements would be to foster teaching staff – student communication, to allow publishing course and laboratory materials, to allow assessing the students knowledge, to offer a support for course laboratory using simulators, to allow student themselves to learn, to practice and to complete self-knowledge assessment problems, and to provide them with a feedback.

The platform described in this paper is incrementally used on course Digital logic at our institution for last several years. To find out how the students perceive this platform, we created three anonymous polls and made it available to this year generation of Digital logic students (winter semester of 2010/2011). We will discuss the obtained findings.

This paper is organized as follows. In Section 2, we describe course elements and give an overview of the platform. In Section 3, we describe laboratory organization and platform support. In section 4, we describe the support for assessment of students' knowledge. In Section 5 we describe support for student learning and self-knowledge assessment. In section 6, we discuss poll results. Section 7 concludes the paper.

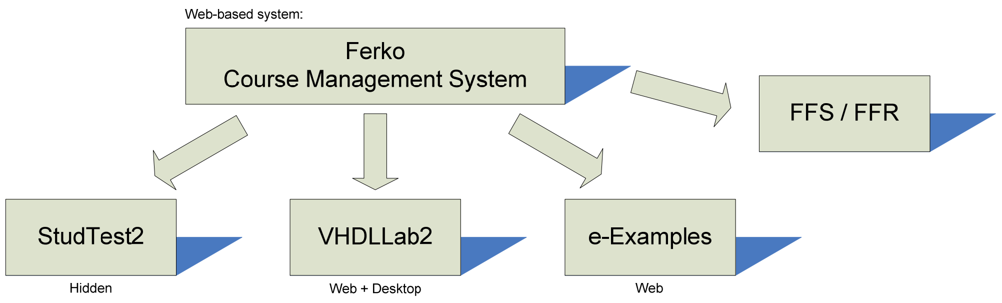


Figure 1. Overview of the integrated platform

II. The Integrated Platform for Digital Logic

The Digital logic course (as currently organized) has following components:

- Lectures (in class),
- Homework assignments,
- Laboratory exercises, and
- Midterm and final exams.

Lectures are given in-class and are complemented by lecture slides. There are 5 homework assignments, each consisting of 10 to 20 problems. Since the average number of students enrolled is between 700 and 800, the requirement on these problems is to be individualized, to discourage students to copy solutions from each-other. Laboratory exercises are VHDL based. During the semester, students should experiment with describing combinatorial and sequential digital logic in VHDL and simulating the resulting circuits. Laboratory exercises range from describing simple combinatorial circuits (such as XOR-gate using AND, OR and NOT-gates) to more complex-ones (such as multiplexer and decoder trees and digital arithmetic circuits), and from simple sequential circuits (such as D, T, SR and JK-flip-flops) to more complex ones (such as counters, timers, and Moore and Mealy automata). On laboratories, students are awarded points by lab instructors and by short exams taken at the end of each lab term. Midterm and final exams are based on multiple-choice tests.

The integrated platform we use is shown in Figure 1. The central component is open-source Java Course Management System (JCMS), the instance of which we installed as *Ferko* System. At our institution, *Ferko* is used by all courses for publishing lecture schedules, midterm and final exam schedules and makeup exam schedules, since it offers centralized calendar tracking facilities for all students. A number of courses which have multiple-choice based exams use *Ferko* to generate answer sheets and publish student results. Scanning and optical recognition is handled by two open source Java-based tools: Free Form Scanner (*FFS* on Figure 1) and Free Form Reader (*FFR* on Figure 1).

Ferko is also responsible for authentication and authorization of students and staff members. Through additional API, this functionality is also offered to other systems which comprise the integrated platform. On course Digital logic, *Ferko* is also used to publish lecture materials (for example, laboratory instructions and various explanations).

III. Laboratory Exercises

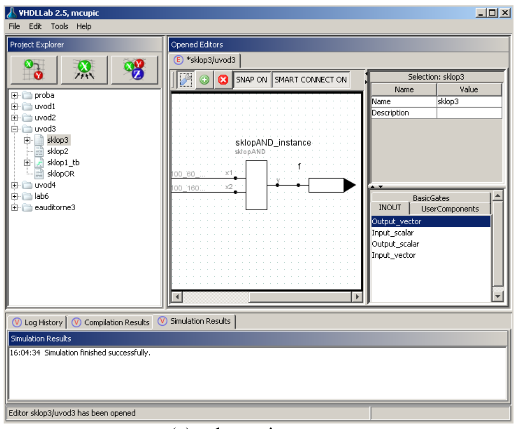
During the semester, we organize seven laboratory exercises divided into three groups. Exercises in first group are devised to allow students to build simple and complex combinatorial circuits from simpler building blocks, using schematics. The exercises range from constructing XOR-gate from AND, OR and NOT-gates to constructing two-part communication system that uses Hamming-codes for message protection. To complete this group of exercises, no knowledge of VHDL is required. Circuits are built by drawing schematics, and later reusing created circuits in higher-level schematics. Using these schematics, students construct testbenches and perform simulations.

Second group of exercises introduce VHDL as language for describing digital logic circuits. Using VHDL, students create behavioral models of multiplexers, decoders and simple single-bit arithmetic unit and then build structural models for multiplexer trees, decoder trees and multi-bit arithmetic unit.

Finally, the third group of exercises is devised to allow students to try to describe memory elements and sequential logic circuits, ranging from simple flip-flops to Moore and Mealy final state automata, again in VHDL.

Since the time scheduled for each exercise is rather limited, we wanted to allow students to prepare themselves for exercises at home, at their own pace and time. In order to do so, we needed a simulation environment that would allow students to complete exercises at home, with a minimal requirement on additional software or licensing issues.

For this reason, we created another open source tool entitled *VHDLLab2*. It is a two-part system, one residing at our server and offering simulation services, user project management and data storage, and the other being a thin client which resides on students computer and acts as an integrated IDE. The client is Java WebStart based application, which requires no installation and can run on any Java-enabled desktop computer. When started, the client connects to the server, obtains students projects and allows the student to manage projects, create various types of circuits (schematics, vhdl-sources, automata, testbenches) and perform simulations. When client needs to simulate a circuit, simulation is performed on server by *ghdl* – a open source simulation tool *VhdlLab2* uses in

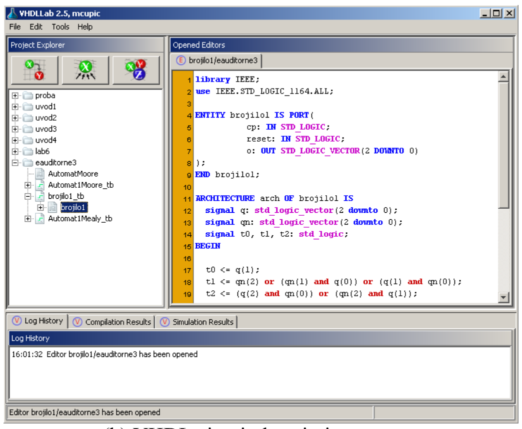


Name	Value
Name	sklop3
Description	

```
16:04:34  Simulation finished successfully.
```

Editor sklop3/uvod3 has been opened

(a) schematic

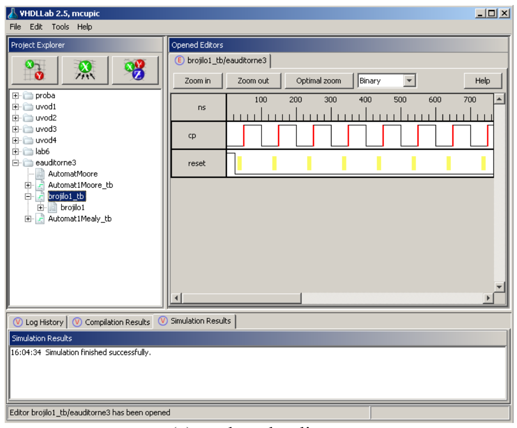


```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

ENTITY brojilol IS PORT(
          cp: IN STD_LOGIC;
          reset: IN STD_LOGIC;
          o: OUT STD_LOGIC_VECTOR(2 DOWNTO 0)
);
END brojilol;

ARCHITECTURE arch OF brojilol IS
  signal q: std_logic_vector(2 downto 0);
  signal qn: std_logic_vector(2 downto 0);
  signal t0, tl, t2: std_logic;
BEGIN

  t0 <= q(1);
  tl <= qn(2) or (qn(1) and q(0)) or (q(1) and qn(0));
  t2 <= (q(2) and qn(0)) or (qn(2) and q(1));
```

```
16:01:32  Editor brojilo1/eauditorne3 has been opened
```

Editor brojilo1/eauditorne3 has been opened

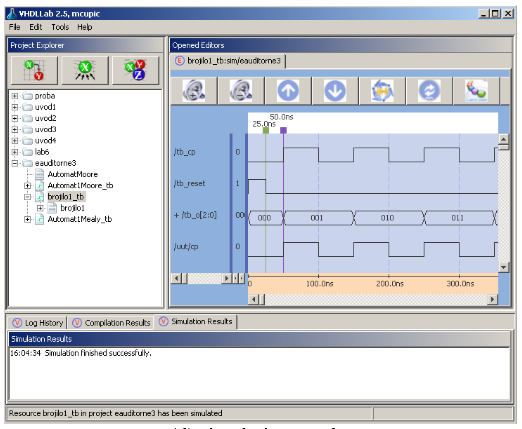
(b) VHDL circuit description



```
16:04:34  Simulation finished successfully.
```

Editor brojilo1_tb/eauditorne3 has been opened

(c) testbench editor



```
16:04:34  Simulation finished successfully.
```

Resource brojilo1_tb in project eauditorne3 has been simulated

(d) simulation results

Figure 2. Several features of VHDLLab2 system.

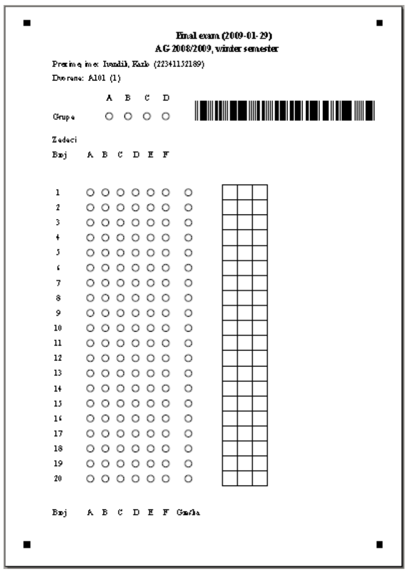
background. Simulation results are then transported back to the client, and are presented to the user. Several screenshots of *VHDLLab2* IDE are presented on Figure 2.

At the end of each laboratory exercise, lab assistant evaluates the students work. At the end of the last exercise in each of the three groups, students knowledge is also examined by a computer based test. The test is typically composed from 10 questions in a period from 10 minutes. Question types are either multiple-choice questions with one (ore more) correct options, or enter-text question, where student must enter answer in some predefined format (e.q. "*Calculate the control bits when using Hamming code with even parity to protect 1111. The answer is?*", and student is expected to enter "111"). To prevent answer copying, question should be as much as individualized as possible. To achieve this, we used *StudTest2* to prepare tests and for the assessment. To students, *StudTest2* system is hidden, since it is wrapped by the *Ferko* system.

IV. Assessment of Students Knowledge

During the semester, students' knowledge is assessed using 6 supervised and 5 unsupervised exams. The three supervised exams that are awarded with rather large amount of points are two midterm exams and one final exam. These exams are composed of 15, 20 and 25 multiple-choice questions, respectively. We typically prepare 4 different groups; each group contains different variant for each question, to make answer copying harder.

Since at our institution the schedule for mandatory exams on all courses is defined and published in *Ferko* at the beginning of semester, we only have to create a room schedule for students, which is also a simple task using *Ferko*'s built-in capabilities. After that, we can define the type of exam to be multiple-choice, and *Ferko* will provide us with PDF document containing blank answer sheets for each student (see Figure 3), ready to be printed.


A B C D
Grupa
Zadaci
Broj A B C D E F
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Broj A B C D E F Greška

Final exam (2009-01-29)
AG 2008/2009, winter semester

Prezime, ime: Ivandić, Karlo (22341152189)
Dvorana: A101 (1)

Figure 3. Answer sheets for exams

When the exams are over, answer sheets are machine processed (using *FFS* and *FFR*), and results along with scans are feed back into *Ferko*. For a generation of 750 students, this means that exams will be processed and results published in no more that two hours after the exam is over.

Other three supervised knowledge assessment are already described – exams at the end of each laboratory group.

Five unsupervised knowledge assessment which are awarded with a rather modest amount of points are homework assignments. The main goal of homework assignments is to encourage students to try to solve assigned tasks themselves, in order for them to see if they understood the lecture matter and if not, to motivate them to reread the lecture notes and other course materials. Each homework is usually open during a period of seven days. To further encourage individual work, problems in homework assignments are individualized, so that the simple solution copying is impossible. For this task, we use StudTest2 server which is charge for creating individualized tasks and evaluating the students' solutions.

V. Learning and Self-Knowledge Assessment

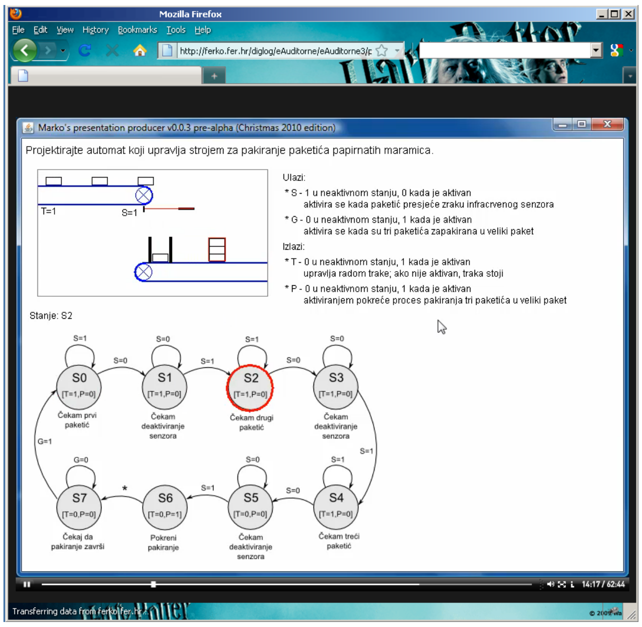
To complement available course materials and mandatory knowledge assessments, we have prepared a set of additional screencasts (entitled *e-examples*) and a set of additional individualized tasks (entitled *e-practice*) which can be accessed by students outside of any mandatory knowledge assessment.

A. e-examples

The semester at our institution is divided into three cycles. For each cycle, we have prepared a set of screencasts in which some parts of theory are explained in a different way than in lectures. Additionally, a number of screencasts explains students how to solve some illustrative problems, by combining the underlying theory and given problem.

First step in preparing each screencast is to prepare screencast materials. This can be, for example, a PowerPoint presentation which will be used as a basis for screencast. However, for this course we wanted to be able to work with presentation in which we could embed rich interactive simulations and animations. For this purpose, we have developed custom Java-based presentation software with support for text and graphics display, and with animation abilities. Complete presentation is described in Java programming language using a series of predefined classes. After the compilation, presentation can be started and navigated chapter-by-chapter or slide-by-slide.

Once the presentation is ready, the taking of screencast must be undertaken. What is required is a good headset with high-quality microphone, and a tool for capturing screencasts, for which there are many available commercial tools, as well as some open source tools. After the capturing is done, there is usually a lot of work needed to edit the screencast, in order to fix the quality of human speaker and to cut out invalid parts. Our experience shows that in order to fully prepare a 30-minute screencast, the total amount of time invested is often more than 5 hours, for a simpler presentation. If the presentation alone is complex, the investment of time can be even much larger.


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://ferko.fer.hr/diglog/eAuditorne/eAuditorne3/
Marko's presentation producer v0.0.3 pre-alpha (Christmas 2010 edition)
T=1
S=1
Stanje: S2
S0 [T=1,P=0] Čekam prvi paketić
S1 [T=1,P=0] Čekam deaktiviranje senzora
S2 [T=1,P=0] Čekam drugi paketić
S3 [T=1,P=0] Čekam deaktiviranje senzora
S4 [T=1,P=0] Čekam treći paketić
S5 [T=0,P=0] Čekam deaktiviranje senzora
S6 [T=0,P=1] Pokreni pakiranje
S7 [T=0,P=0] Čekaj da pakiranje završi
S=1
S=0
G=1
G=0
*
14:17 / 62:44
Transferring data from ferko.fer.hr...

Projektirajte automat koji upravlja strojem za pakiranje paketića papirnatih maramica.

Ulazi:
* S - 1 u neaktivnom stanju, 0 kada je aktivan
  aktivira se kada paketić presječe zraku infracrvenog senzora
* G - 0 u neaktivnom stanju, 1 kada je aktivan
  aktivira se kada su tri paketića zapakirana u veliki paket

Izlazi:
* T - 0 u neaktivnom stanju, 1 kada je aktivan
  upravlja radom trake; ako nije aktivan, traka stoji
* P - 0 u neaktivnom stanju, 1 kada je aktivan
  aktiviranjem pokreće proces pakiranja tri paketića u veliki paket

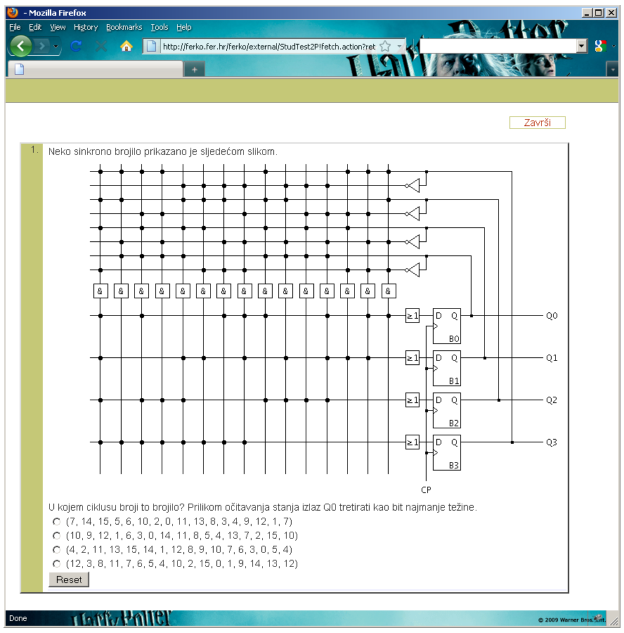
Figure 4. Screencast example for teaching Moore-type automata

Example of such a screencast is shown on Figure 4. It is a simple industrial plant with two conveyor belts, a sensor and the packed-packing machine, for which a controller must be devised. The upper left box visualizes the plant state while the lower part of the slide contains developed automata (current state is presented by red circle). Both images are animated and mutually synchronized. Presenter can pause the simulation or reset it any time during the capturing of screencast in order to be able to better explain what is happening.

B. e-practice

To provide students with additional means to practice and to perform self-knowledge assessment, we have used wiki subsystem which is offered as part of *Ferko* for each course. Using wiki pages, each course can easily build a number of web pages on various course topics. Since integrated wiki allows customization, we implemented a connector to the *StudTest2* server, which allowed us to offer to the students a direct access to each prepared individualized problem. Using this, each student could access each of the tasks as many times as he/she wished, solve it and receive the feedback. Since the problems are individualized, each new access would generate a different variant of the problem. We prepared 57 problems for the first lecture cycle, 48 problems for the second lecture cycle and 42 problems for the third lecture cycle.

An example of problems we made available to students is shown on Figure 5. Each time student accesses such a problem, a new selection of flip-flop will be made, and a new circuit counting in some randomly generated cycle will be created (and drawn on image). Students task is to discover in which cycle the counter counts, thus allowing the student to verify his/hers analysis capabilities. From the problems we made available to students, some of them test students recall, some test students analysis and some students synthesis abilities.


- Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://ferko.fer.hr/ferko/external/StudTest2P!fetch.action?ret
Završi
Q0
Q1
Q2
Q3
B0
B1
B2
B3
CP
Reset
Done
© 2009 Warner Bros. Ent.

1. Neko sinkrono brojilo prikazano je sljedećom slikom.

U kojem ciklusu broji to brojilo? Prilikom očitavanja stanja izlaz Q0 tretirati kao bit najmanje težine.

- (7, 14, 15, 5, 6, 10, 2, 0, 11, 13, 8, 3, 4, 9, 12, 1, 7)
- (10, 9, 12, 1, 6, 3, 0, 14, 11, 8, 5, 4, 13, 7, 2, 15, 10)
- (4, 2, 11, 13, 15, 14, 1, 12, 8, 9, 10, 7, 6, 3, 0, 5, 4)
- (12, 3, 8, 11, 7, 6, 5, 4, 10, 2, 15, 0, 1, 9, 14, 13, 12)

Figure 5. Example of individualized problem available in *e-practice*

VI. STUDENTS OPPINION

In order to evaluate various aspects of developed platform, at the end of current academic year we offered three anonymous polls in form of web-based questionnaires. The students had to log-in on *Ferko*, and then they could open available questionnaires. The system only tracks if student completed the poll or not, so the anonymity is guarantied.

We offered three questionnaires. The first one was about laboratory exercises and homework assignments and contained 31 questions (several of which were also open ended questions). The second questionnaire was about e-examples and contained 14 questions. The third questionnaire was about e-practice and also contained 14 questions.

Due to limited amount of space available in this paper, we will present only a selection of 5 most general questions, for which short version of text as results are presented in Figure 6.

Questions were, as follow. Q1: "*Did homework assignments help you to better learn the course material?* (1: strongly disagree, 5: strongly agree)", Q2: "*Did homework assignments help you to better prepare for exams?* (1: strongly disagree, 5: strongly agree)", Q3: "*If each student would get exactly the same questions in homework, would you be equally motivated to solve problems by yourself?* (no, then I would copy solutions from others / yes, I would still try to solve problems myself)", Q4: "*Do you find e-examples to be useful?* (1: strongly disagree, 5: strongly agree)" and Q5: "*Did existence of e-practice enable you to better learn the course matter?* (1: strongly disagree, 5: strongly agree)".

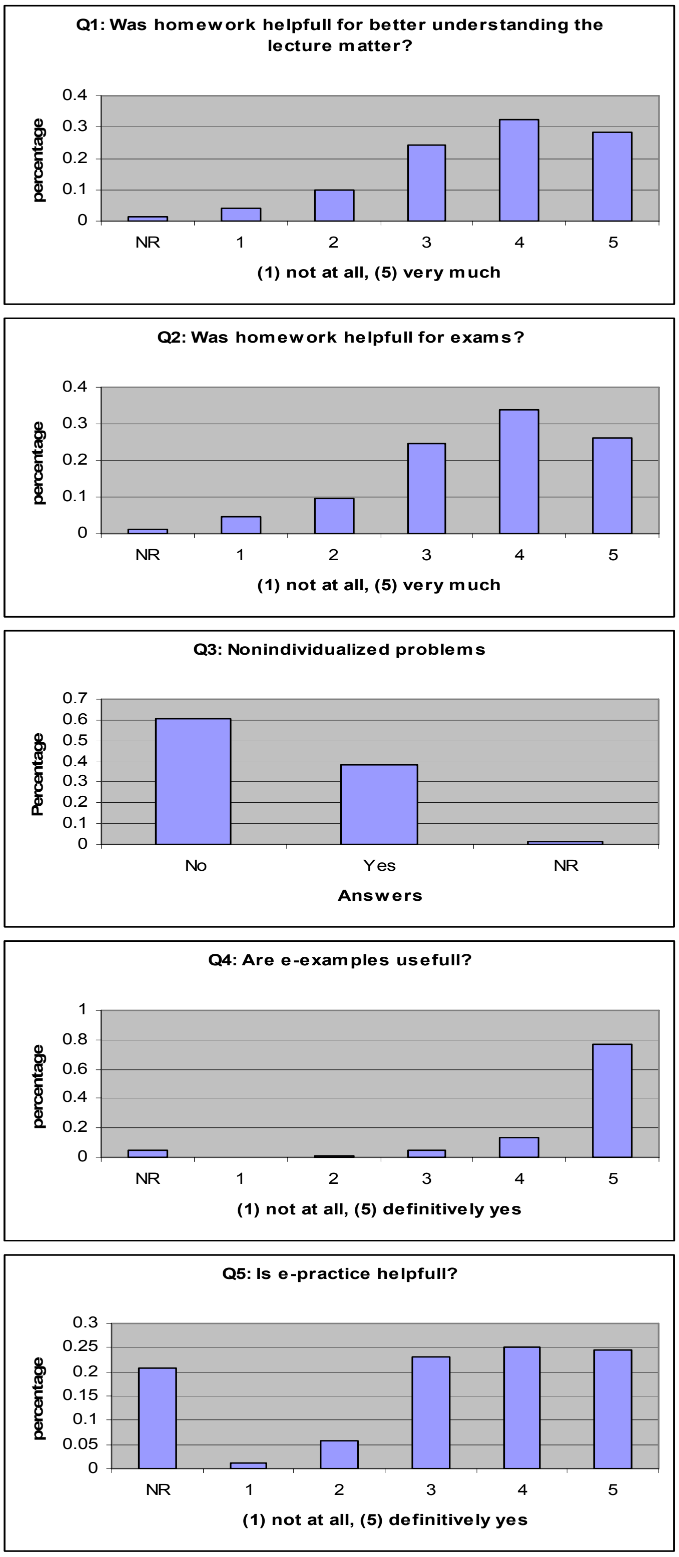


Figure 6. Selection of poll results

Question Q1, Q2, Q3, Q4 and Q5 were answered by 377, 378, 374, 360 and 344 students, respectively. Total number of students in generation was 776. Option NR in Q1, Q2, Q4 and Q5 means "did not use it / solve it".

Q1 had average of 3.71 ± 1.10, Q2 had average of 3.68 ± 1.10, Q4 had average of 4.74 ± 0.60, and Q5 had average of 3.83 ± 1.00. These results are all significant at p=0.01 level of significance. While questions Q1 and Q2 indicate that students do positively believe that individualized problems prepared in StudTest2 and offered in form of homework assignment helped them to better learn the course matter and to better prepare themselves for exams, it is interesting to observe that allowing them to additionally choose which problems to solve, when to solve them and how many times (which is the essence of question Q5) received even higher score.

Also, as the results for the question Q4 clearly indicates, students found *e-examples* in form of screencasts very usefull. By analyzing web server logs, we determined that in period from the January, 2, 2011 to February, 5, 2011 screencasts were downloaded 10155 times. What is interesting is to note that peeks in accessing always occurred in periods close to exams (final exam and makeup exam).

It is also interesting to note the results for the question Q3. Over 60% of students answered that if the problems in homework assignments would not be individualized, they would be more inclined to simply copy the solutions from others, instead to try to solve the problems themselves.

After the analysis of poll, the general impression is that the developed platform is well accepted and that the students feel that the usage of described tools and services indeed can help them to better learn the course matter.

VII. CONLUSION

In this paper an integrated platform for teaching and learning of digital logic is described. Speaking from the teaching-staff side, using this platform helped us in many different ways. For start, using the *Ferko* system helped us to reduce the burden with course organization and time management, and simplified problem tracking and communication with students.

But what is most important, it lent itself as a platform for constructing and connecting other services. This allowed us to start creating new tools and services with the goal of enabling students to achieve better learning outcomes. An example is usage of *FFS* and *FFR* tools which allowed us to fully process, score and publish exam results for 800 students in less then two hours after the exam finished. Also, the integration of *StudTest2* system and *VHDLLab2* system significantly contributed to improving students' quality of learning, especially since it allowed them to solve laboratory exercises, to practice and to do self knowledge assessments from home, at their own time and pace. The inclusion of wiki subsystem allowed us to more easily create course related web pages and to provide better learning materials.

As the poll results shows, the project of creating integrated platform was indeed well accepted by the students, and many believe that they benefited from it.

What remains for the future is to explore how to further utilize this platform to create even richer set of tools and services that would provoke even more interest from the students, and that would lend itself to a better and more natural way of learning.

REFERENCES

[1] J.C. Lindenlaub, "Restructuring the teaching of digital logic design: A progress report," in Proceedings 23th Annual Conference Frontiers in Education (FIE 1993), 1993, pp. 192

[2] S. Wood, and R. Danielson, "Java-based instructional materials for introductory logic design courses," in Proceedings 30th Annual Conference Frontiers in Education (FIE 2000), 2000, Vol. 2, pp. S2D/10 - S2D/16

[3] T. Weng, Y. Zhu, and C.K. Cheng, "Digital design and programmable logic boards: Do students actually learn more?," in 38th Annual Frontiers in Education Conference (FIE 2008), 2008, pp. S1H-1

[4] Y. Zhu, T. Weng, and C.K. Cheng, "Enhancing Learning Effectiveness in Digital Design Courses Through the Use of Programmable Logic Boards," in IEEE Transactions on Education, Feb. 2009, Vol. 52, Issue 1, pp. 151 - 156

[5] M. Hessami, and J. Sillitoe, "The role of laboratory experiments and the impact of high-tech equipment on engineering education," in Australasian Journal of Engineering Education, 3, pp. 119-126.

[6] M. Johnson, and B. Craig, "Computer systems pedagogy using digital logic simulation," in Proceedings. International Conference on Computers in Education, 2002., pp. 703-704

[7] M. Serra, E. Wang, and J.C. Muzio, "A multimedia virtual lab for digital logic design," in IEEE International Conference on Microelectronic Systems Education (MSE '99), 1999, pp. 39

[8] P. Jamieson, "Scotty in the Engine Room - A Game to Help Learn Digital System Design and More," in Meaningful Play 2010 Conference Proceedings, Michigan State University, USA, October 21-23, 2010.

[9] A.N. Greca, J.K. Harris, and S.E. Butler, "Work in Progress – Integrating Student's Research Project into the Digital Logic Design Course to Enhance Learning," in Proceedings 35th Annual Conference Frontiers in Education (FIE 2005), 2005, pp. S1H

[10] G.L. Herman, and J. Handzik, "A preliminary pedagogical comparison study using the digital logic concept inventory," in 40th Annual Frontiers in Education Conference (FIE 2010), 2010, pp. F1G-1 - F1G-6