Experience of Blended Learning in the Subject - Architecture of Computers

Michael Dolinsky
Faculty of Mathematics and Programming Technologies, Francisk Skorina Gomel State University, Belarus
E-mail: dolinsky@gsu.by

Abstract

The proposed study discusses about the real-time experience of blended teaching for the students of the Faculty of Mathematics and Programming Technologies in the subject "Computer Architecture" based on the use of distance learning instrumental system DL.GSU.BY and special software complexes developed at the F. Skorina State University under the guidance of author. Special software packages include: HLCCAD - software system for designing, modeling and debugging the functional architectures of digital devices; Winter, a software system for developing and debugging programs in assembler and C for various microcontrollers; C-MPDL – firmware description language for computer architecture components; constructor of training and control flash tasks. The proposed method of conducting lectures and practical classes, as well as the organization of students' independent work and an assessment monitoring system that orients students toward a permanent increase in the assessment and quality of knowledge, skills, and abilities throughout the academic semester, are described.

Keywords: Blended learning, computer architecture, instrumental system of distance learning, high level chip computer aided design, program debugging

1. Introduction

Due to the recent pandemic situation like COVID-19, the effectiveness of online learning needs to be improved [1-3, 10, 11]. One of the areas of such work is the development of an online learning environment [4, 5]. Another potential research direction is the use of blended learning [6, 7, 9, 12, 13], when the shortcomings of online learning are offset by the work of the teacher with students, as well as the interaction of students with each other during classes in classrooms. The problems are supranational [8]. The introduction of new information technologies in traditional education leads to a significant increase in the

DOI: https://doi.org/10.36548/jitdw.2022.3.004
Received: 18.07.2022, received in revised form: 27.07.2022, accepted: 18.08.2022, published: 23.08.2022

quality and intensity of the educational process. This paper presents the experience of such blended teaching of the subject "Computer Architecture" at the F. Skorina GSU for students of the specialties "Informatics and Programming Technologies", "Information Technology Software", "Applied Informatics". During training, the DL site http://dl.gsu.by and a set of special software systems are developed.

2. Research Questions

Within the framework of the subject "Computer Architecture" it is studied how the development of Intel processors (starting from 8080) and non-Intel representative architectures, such as: digital signal processors, transputers, relational associative processors (or database processors), computers are controlled by data flows, multiprocessor systems and supercomputers.

The main problems that need to be addressed in teaching this course are:

- High-quality assimilation of theoretical material by students
- The formation of students' practical skills in the analysis and synthesis of fragments of computer architectures
- Increasing motivation to study different levels of training. The following is the proposed approach to solve the aforementioned challenges.

3. Methodology

3.1 Lecture classes

For the convenience of studying and teaching, the theory on the subject is immersed in DL in such a way that all its sections are available to students when studying any topic. The same materials are used when conducting lectures using a laptop and a multimedia projector. In addition, the lecture hall is equipped with wireless access points to the university computer network and the DL site, as well as power connection points. This allows students to use their own laptops to view and analyze the information offered by the lecturer. Today, the situation of using a laptop by almost every student is typical. In rare cases, there will be one laptop for two students sitting in the same table. In fact, the lecture hall turns into a mobile computer class. This gives completely new opportunities for lectures as described below.

Individual work with theory: Students can refer to any theoretical segment from their laptop as per their requirement, both during the lecture by the teacher and after it is completed.

Individual transition to practice: Since practical tasks are also performed in the DL system, students who have successfully mastered the theoretical material presented by the lecturer can proceed to the implementation of practical exercises on the topic of the current lecture.

Re-explaining the material: The possibility of an individual transition to practice helps to explain complex material to less prepared students repeatedly or even three times, without becoming detrimental to students, who have already mastered the material. The repeated explanation of the material can be performed both by a teacher for a certain group of students, and a student for a table neighbor. In addition, the lecturer, if necessary, consults students, who perform practical tasks during the lecture.

Activation of lectures: DL provides automatic verification of the correctness of the execution of tasks. If the task is completed incorrectly, the student will be allowed to complete it again. In addition, the DL system keeps track of the number of tasks completed by each student (or a team of two students). For each lecture session, each student is awarded bonus points, which is proportional to the number of tasks solved at the lecture. If 2 students worked on one laptop, the bonus will be divided into half. Bonus points earned by a student in a lecture will significantly affect his/her examination grade. This approach stimulates students both to accelerate the assimilation of the new theory proposed at the lecture, and to attain its maximum practical consolidation by directly solving as many problems as possible during the lecture.

Automated personalized learning: The task folder for each lecture contains subfolders, named "Training" and "Control". The "Control" folder contains the tasks that check the quality of assimilation of a new topic. The "Training" folder contains a tree-like system of tasks leading to the solution of each task. Thus, each student can receive a personal learning path depending on the level of training, motivation and psychophysical state at the time of learning.

Forum DL: plays an important role in the organization of lectures. For each academic discipline, a special topic is created in the forum. The first messages in the topic contain a program for studying the discipline connected to materials for each topic. Subsequent messages are of the following types. Announcement of the upcoming lecture - usually laid

out at the end of the academic week by including a plan for the upcoming lecture, links to materials on the topic of the lecture and recommendations for students in order to get prepared for the next lecture. After each lecture, each student has the right to write in the same place about what seemed incomprehensible to him/her at the lecture or to make suggestions on the form and organization of classes. Finally, the teacher can write after each lecture about his/her feelings, joys and concerns regarding the current state of the educational process.

3.2 Workshops

In practical classes, a student can perform one of the following types of work: training, test, control section.

Weekly check-ups: With automatic verification of solutions in the DL system is considered as a key component in the organization of practical classes. The duration of the tests is about two hours (a couple, a break before it and a break after it). At this time, special tasks (10 pieces or more) are opened and its solution will be evaluated by the number of fully completed tasks. Assignments will open 15 minutes before the commencement of class, so that the interested students can start working without waiting for a call to the class. Assignments are closed 15 minutes after the completion of the pair so that students have the opportunity to “finish off” the assignment, for which, “one minute is not enough”. For assessment, all the students will be offered with the same task, so that the students can discuss the answers after the examination or even directly during its execution.

At the same time, the DL system maintains a strict control so that each student completes all the tasks independently. To do this, when performing a test, each student is required to enter the university network under a special OLYMP account, which has access only to the DL site and the OLYMP folder of the D: drive of their machine, which forces the student to do all the tasks "with his/her own hands". In case of violation of the rules, the DL system blocks the possibility of sending decisions by the student-violator.

The teacher controls the ability of students to access educational materials located on the DL: theory, forum, and additional learning tasks. At the request of the teacher, these materials may or may not be available during the test. All solutions are checked automatically in the DL system. In the process of performing tests, each student can always see the current table of test results of all students.

Each test includes both the assignments on the theory studied at last lecture and assignments for the entire course being studied (including those on topics that have not yet been lectured). This approach provides both control over the assimilation of the material and stimulation for advanced learning. In order to stimulate an active work in the learning process, each student has access to information about all the tasks performed by all students of the group present in class since the beginning of the school year, ranked from best to worst.

Control slices: This is a type of control work in which each student is offered with his/her own set of exactly 10 tasks. The number of tasks solved in a couple of tasks will fundamentally affect the final examination grade of the subject - it cannot be more than the grade for the control section. The control section is written once a week - according to the readiness and desire of the student. The control slice can be written unlimited number of times. The best score is then gets fixed. If the current estimate is less than the fixed one, it will be ignored. This approach stimulates the desire of students to repeatedly write a control section during the semester in order to obtain the highest possible grade.

3.3 Independent work

The entire education system, including lectures and practical classes, is aimed at preparing the student to work independently as early as possible in terms of gaining knowledge from lectures and practical classes, as well as in individual self-training outside of class. One of the important advantages for students is to work independently on individual tasks. In individual tasks there are tasks, starting with the simplest ones and on the other hand, individual tasks are most diverse in type, form and subject. In addition, each credited individual task contributes more to the automatic rating of the student's knowledge, for example, a test task or a study task. However, in order to avoid cheating, each individual task is credited only to the student who initially passes this task. And, finally, in each topic, each student is credited with only one task. It is important to note that many tasks in the subject were created by the students themselves. This is due to the fact that this type of independent activity is highly evaluated.

3.4 Assessment automation

The current assessment status of all students is available to everyone from the first day of class. The main factors considered in this assessment are theory, practice, training, individual assignments, and development of own assignments. For each of these activities, a

student can generally earn a grade of up to 10 or more. A sum of these marks divided by 5 is the base mark obtained in the exam.

The column "bonuses" works towards increasing the rating. Downward – column “Omissions”. Currently, the scale of punishment for omissions "in geometric progression" is adopted: 1 point is deducted for 1 pass (to compensate, it is enough to solve one individual task), for 2 passes - 2 points, for 3 passes - 4 points, etc. for K passes, $2^{(K-1)}$ points are subtracted. A system in which the student forms an assessment on the subject being studied to a large extent encourages interaction and a creative position in the study of the material.

3.5 Control tool for classes missing

The student may have objective reasons for missing classes, and on the other hand, inconvenience experienced with the teacher and regular absences from classes can lead to a significant deterioration in the quality of education. In the described system, this problem is solved by developing an automated system for recording the absenteeism and working off them. At the beginning of the semester, the dates of all classes for the semester appear in the Class Absences folder.

During the class (or shortly after), the teacher issues passes to all students who are absent from class. Each student at any convenient time can enter the DL system, work for an hour and a half (the net time of one lesson at the university) (as evidenced by the entries in the protocol for automatic verification of decisions) and indicate the date of the pass that he/she worked out. The entry of a student from such a group completed such a lesson, worked from such time to such time, and did so many tasks will automatically appear in a special forum topic.

A link in the same entry to the protocol of the student's work will be established during this time period. The teacher examines the protocol and decides whether or not to accept the pass. Work may not be accepted if the student attempts to "pretend" to work. In the case of acceptance of mining, the pass will be reset. The working off rules have been implemented to ensure that students do not put off the working passes indefinitely. It can be worked out in pairs without the presence of a teacher within a week of receiving the pass. After that, the student must either work out in the presence of a teacher or work out with an increase in working time: for each "extra" week between the pass and working off, an extra hour and a half should be spent.

3.6 Software Tools

3.6.1 HLCCAD software system

HLCCAD software system (High Level Chip Computer-Aided Design, see picture 1) is intended to achieve an efficient development of hardware for functionally complex digital systems.

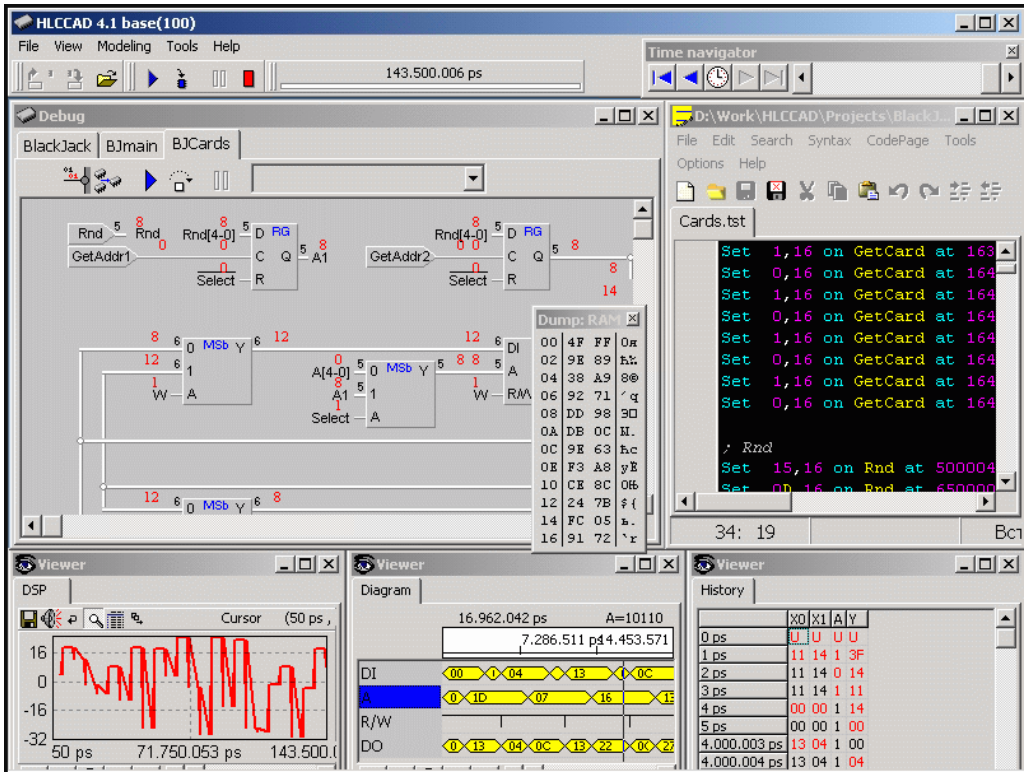

Figure 1. Debugging digital circuits in the HLCCAD system

The HLCCAD system includes a library of standard digital circuit design components by including the logical elements: AND, OR, XOR, NOT; combinational circuits: adder, decoder, encoder, multiplexer; memory elements: flip-flops, registers, RAM, and ROM. HLLCAD supports the hierarchical design, which allows to implement the previously developed schemes in new schemes. To check the correctness of circuits, their simulation will be used. By specifying input actions interactively from a file, the user can analyze the output values, as well as the values at any inputs/outputs within the circuit at any time. After debugging the circuit, the student submits it for verification. The DL system receives the corresponding file and launches HLCCAD to simulate the operation of the circuit on the tests prepared by the task author. The result of the performed check is reported to the student within few minutes.

3.6.2 Winter software system

Winter software system intends to incorporate the development and debugging of assembler and C-programs for various microcontrollers (see figure 2).

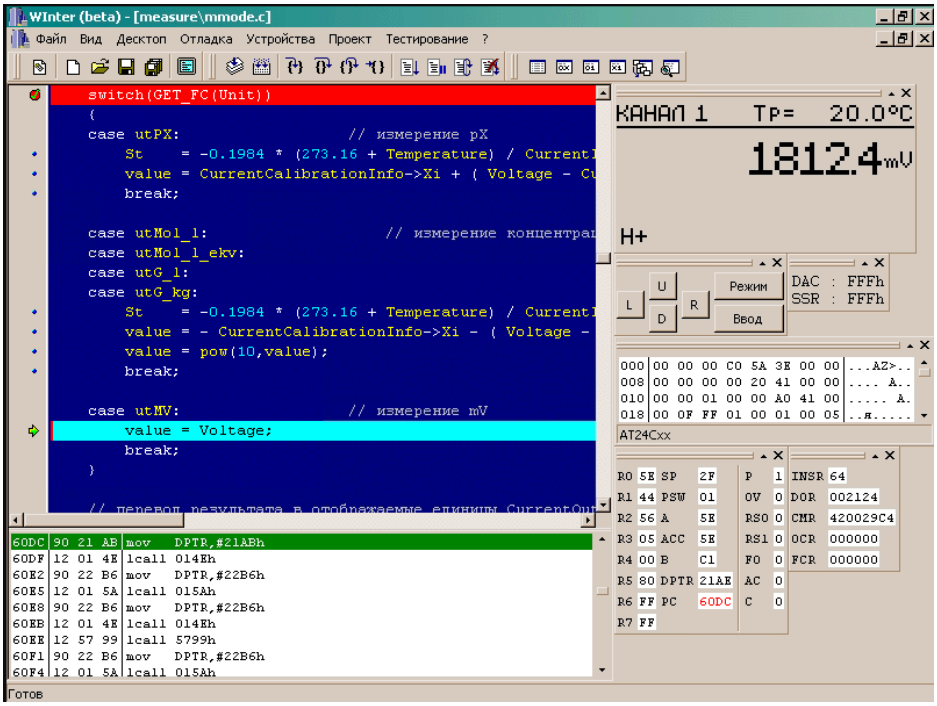

Figure 2. Debugging programs in Winter

Currently, some models of microcontrollers from Intel, Motorola and Texas Instruments as well as models for MPDL and VHDM pseudo-processors have been created. By using the winter system, the user can type and edit programs before executing and debugging it. As usual at the debugging phase, the user can observe the values shown in registers and variables during execution. It is interactively possible to change the values of variables and registers.

3.6.3 Microprogramming language C-MPDL

A distinctive feature of the proposed approach is the utilization of the C-MPDL microprogramming language, which is developed under the guidance of the author and the program debugging environment that allows generating a microprogram automaton with rigid logic for a debugged microprogram in order to execute the algorithm of this microprogram. In the Winter environment, debugging of such C-MPDL programs is provided, and in the HLCCAD environment, the generation of microprogram automata from C-MPDL programs is provided.

a) Data types: The following data types are supported:

int - signed integer,
dimension - arbitrary;
unsigned - unsigned integer, dimension - arbitrary;
double - floating point, dimension - 64 bits;
one-dimensional arrays of the enumerated types

When generating a hardware circuit of a device, it is convenient to specify the dimensions of variables and external contacts. For this, the type system of the C language has been extended. When declaring an integer variable, its dimension can be specified. The dimension can be arbitrary, for example 8 bits, 5 bits, 1 bit, 500 bits. If the dimension is not specified explicitly, then the variable will have a dimension of 32 bits.

When evaluating the expressions containing variables of different dimensions, the following type casting rules are used: the result of an operation on two operands has the dimension of larger operand; the result of an operation with signed (int) and unsigned (unsigned) operands is signed (int).

To describe external contacts, the specifiers __in (input), __out (output), and __inout (bidirectional) are used in the process of variable declaration.

An example of a description of variables indicating the dimensions and types of contacts are:

```
int __in __bits(8) in, __out __bits(3) out;
unsigned __bits(1) bit_array[32];
```

The variable in has a dimension of 8 bits and is considered as the input of the device, out is 3 bits and is considered as the output. The bit_array consists of 32 elements, each of 1 bit in size.

b) Constants: Integer, character and floating point constants are also supported.

Character constants are enclosed in single quotes, for example, 'A' is the character 'A' or '\x10' is the character with hexadecimal code 10.

Integer constants are 32 bits. As in the C language, you can specify the number system of integer constants using prefixes:

- octal if the number starts with '0', for example, 0777 is 777 in base 8;

- hexadecimal - start with prefix '0x' or '0X', eg 0xff10;
- decimal - start with any digit except zero.
- Unsigned constants use the 'U' or 'u' suffix, for example 400U.

The syntax for floating point constants is similar to C, for example:

1.23121e-5
.234
10.0

c) Operators: The following are supported

Execution control operators:

for
if ... else
while
do ... while
break, continue
switch
calling a function with arguments, returning a value from the function

Arithmetic and logical operators:

=
+, -, *, /, %
&, |, ^, ~
<<, >>
+=, -=, *=, /=, %=, <<=, >>=, &=, |=, ^=
==, !=, <, <=, >, >=, &&, ||, !
++, --
? (select statement)
operator ',' (comma)

The syntax and purpose of the listed operators is completely analogous to the C language.

d) Functions: When using a sequential architecture, a stack is usually utilized to call a function based on which the return address and arguments are placed. In the architecture of

microprogram automata, this approach remains inefficient. First, this architecture does not have an addressable code memory, and therefore a return address gets easily stored. Second, the stack remains as a single addressable memory that cannot be efficiently access in parallel manner.

To call a function, special registers are used, wherein the arguments are placed and a register in which the return point from the function is stored. The functional operators operate based on these registers. This imposes restrictions - firstly, a recursive function call is impossible, and secondly, a call to the same function cannot be performed in parallel blocks. The last limitation is caused not only by the absence of a stack, but also by the internal structure of the control automaton.

e) Lack of pointers: In the architecture of microprogram automata, there is no single addressable memory; instead, there are many registers of arbitrary dimension, which can be accessed in parallel manner. An array is a single register accessed by element index. Accesses to the same array cannot be performed in parallel. When writing a program for such an architecture, it is not efficient to use pointers that are available in the standard C language, as they are not supported in the compiler.

f) Flash task constructor

Methods of work of the trainee when performing the designed tasks:

- Drag and drop images to the desired position with the mouse.
- Drawings can be not only dragged but also rotated by using the left arrow and right arrow keys.
- Tasks that are performed by using mouse clicks on the desired area.
- A task to obtain the correct result, as it is required to “flip through” the possible answers in search of the correct one with mouse clicks. The task is considered as completed if the correct answers are selected in all fields.
- Coloring some areas by using the standard "brush cursor dip" in the desired paint.
- Connection of several points (by successive mouse clicks at these points).
- Keyboard input.

g) Principles of designing single tasks: A single task is constructed from arbitrary drawings (ready-made or created directly in the constructor), inscriptions, and special active elements

of the constructor such as: tables, lines, enumerable fields, input fields, timer, and lives. The result of task construction is a text file in XML format, which fully describes the task, all its components and their settings accompanied by the files of used external drawings (in jpeg, png formats). When this task is completed, a special program "Player" is launched to read the corresponding XML file and the used drawings, which presents the task to the student. In each task, the "First" button automatically appears by allowing the student, who is confused when completing a difficult task, to return to his/her original state. This button can also be used by the teacher - if he/she (or another student) helped someone to complete the tasks - to force the task to be repeated again. To further complicate the task, the author of the task can include a special button "Check" in the task. If this button is available, the correctness of the task is checked only at the moment when the student presses this button. If the "Check" button is missing, the correctness of the job is checked continuously.

h) Principles of designing task packages: Tasks can be combined by the constructor into packages. The package of tasks can be either controlling or teaching. In the case of a controlling package, tasks have a linear structure and are issued sequentially. The transition from one task to another occurs in one of two cases: the task is completed correctly or the time allotted by the author to complete this task has expired. The purpose of the control package is to assess the level of preparedness of the student in a particular topic. The result is composed of the number of correctly completed tasks from a certain number of proposed tasks.

The training package of tasks has a tree structure. Tasks of the next level nesting are training tasks based on the current task. If the student has completed the task correctly, then all training tasks will be skipped, otherwise he/she will be provided with first lead-in task, if it is completed correctly, the second lead-in task will be continued. After the correct completion of the last lead-up task, the student is presented with the initial incorrectly completed task. The student will be provided with two additional buttons to manually navigate through the tasks “I don’t know” and “I understand”. In each lead-in task, the author can place his/her lead-ins, etc.

i) A note about the obsolescence of flash technology: The flash task constructor has been developed since 2010, and a huge number (thousands) of training and control tasks have been completed with its assistance. Currently, Flash technology is not standardly supported by many browsers. At the same time, in order to use flash tasks, there are many ways to enable flash support, which is done in classrooms and on students' computers. For security reasons,

students are advised to work only on DL in such browsers. We also started developing an HTML5 player (based on an XML file with a task description) to abandon the requirement of supporting flash in browsers to complete the tasks relevant to the subject.

3.7 Types of Practical Tasks

The following are examples of different types of tasks used in the learning process.

3.7.1 Hardware Design of Architecture Fragments

Implement the BSR instruction of the Intel 80386 processor in hardware.

BSR oper1, oper2.

Reverse scanning bits of the 2nd operand and entering the number of the 1st unit in the 1st operand. Also, ZF=1 if oper2=0, otherwise ZF=0.

Oper2Oper 1ZF	16161	input output output

Example:

Oper2: 1192 dec

Oper1: Dec 10

ZF: 0

Note that only solutions developed manually in the form of a diagram in the HLCCAD software system are automatically accepted for verification.

3.7.2 Microprogram design of fragments of architectures

Implement the BSR command of the Intel 80386 processor in firmware.

BSR oper1, oper2.

Reverse scanning bits of the 2nd operand and entering the number of the 1st unit in the 1st operand. Also, ZF=1 if oper2=0, otherwise ZF=0.

Oper2Oper 1ZF	16161	input output output

Example:

Oper2: 1192 dec

Oper1: Dec 10

ZF: 0

Note that only solutions developed in the form of a C-MPDL program and then generated in the HLCCAD software system in the form of a microprogram automaton scheme are automatically accepted for verification.

3.7.3 Development of assembler programs for various architectures

From a four-digit integer, get a new number consisting of odd digits. The task indicates the assembler of which processor it needs to be solved. Recall that assemblers of the following processors are supported: Intel 8051, Intel 8086, Atmel AT90S2313, Atmel AT90S2323, Motorola M68HC05 and Motorola M68HC08, TMS370.

3.7.4 Creation of new tasks for hardware/firmware design of architecture fragments

The student develops the conditions of the problem, tests and the author's solution, after which the task is immersed in the system DL.

3.7.5 Training and control flash tasks of various types on the topics of the lecture

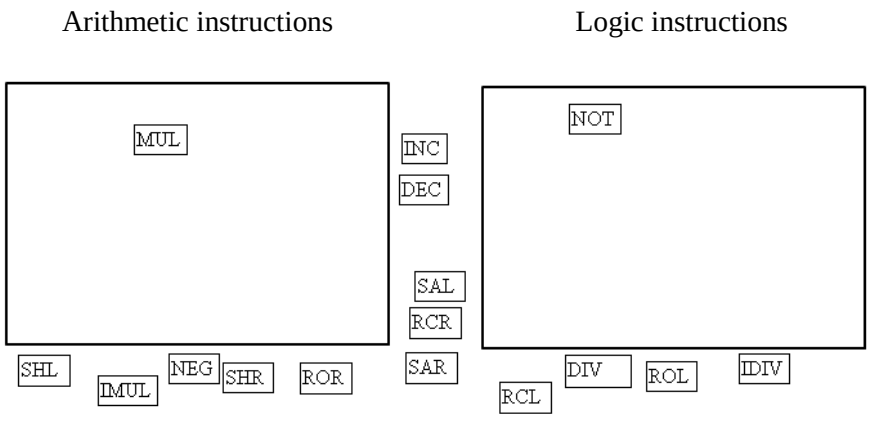

Figure 3. Task to check the understanding of the meaning of assembler instructions

Task presented at figure 3 requires to spread the mnemonics of assembly instructions into the appropriate areas - arithmetic / logical instructions.

3.7.6 Creation of new training and control flash tasks of various types on the topics of the lecture

The student develops a flash task, after which it is immersed in the system DL.

4. Discussion

Many years of experience in the practical application of the aforementioned methods and means of teaching students have showed a significant improvement in the quality of learning process by increasing the students' motivation for learning, adaptive personal learning, and a combination of individual and collective learning. Particularly, the usefulness of the software tools is developed under the guidance for modeling and debugging the functional circuits and programs for microprocessors.

5. Conclusion

The proposed research study presents a novel method for integrating traditional and online learning by efficiently utilizing the DL site and special software tools for modeling and debugging the functional diagrams of digital devices and programs for microprocessors. Integration with DL provides an automation in the issuance of theory and assignments to students, as well as verification of solutions. The final result is an evident for the significant increase in the quality and intensity of the learning process, when working in classrooms as well as working at home.

References

[1] Akyol, Zehra, and D. Randy Garrison. "Assessing metacognition in an online community of inquiry." *The Internet and higher education* 14, no. 3 (2011): 183-190.

[2] Bateman, Tiffani S. "Using Academic Social Networks to Enhance the Student Experience in Online Education." *Online Learning* 25, no. 4 (2021).

[3] Chen, Ye, Jing Lei, and Jiaming Cheng. "What if Online Students Take on the Responsibility: Students' Cognitive Presence and Peer Facilitation Techniques." *Online Learning* 23, no. 1 (2019): 37-61.

[4] Cleveland-Innes, Marti, and Prisca Campbell. "Emotional presence, learning, and the online learning environment." *The International Review of Research in Open and Distributed Learning* 13, no. 4 (2012): 269-292.

[5] Garrison, D. R. "E-learning in the twenty-first century: A community of inquiry framework for research and practice." (2017).

[6] Halverson, Lisa R., Kristian J. Spring, Sabrina Huyett, Curtis R. Henrie, and Charles R. Graham. "Blended learning research in higher education and K-12 settings." *Learning, design, and technology* (2017): 1-30.

[7] Harrell, Kyleigh B., and Jillian L. Wendt. "The impact of blended learning on community of inquiry and perceived learning among high school learners enrolled in a public charter school." *Journal of Research on Technology in Education* 51, no. 3 (2019): 259-272.

[8] Moreira, Jose, Antnio Ferreira, and Ana Almeida. "Comparing communities of inquiry of Portuguese higher education students: One for all or one for each?." *Open Praxis* 5, no. 2 (2013): 165-178.

[9] Pool, Jessica, Gerda Marie Reitsma, and Dirk Nicolaas Van den Berg. "Revised community of inquiry framework: Examining learning presence in a blended mode of delivery." (2017).

[10] Schunk, Dale H., and Barry J. Zimmerman, eds. *Motivation and self-regulated learning: Theory, research, and applications*. Routledge, 2012.

[11] Swan, Karen. "Learning effectiveness online: What the research tells us." *Elements of quality online education, practice and direction* 4, no. 1 (2003): 13-47.

[12] Vaughan, Norman, and D. Randy Garrison. "Creating cognitive presence in a blended faculty development community." *The Internet and higher education* 8, no. 1 (2005): 1-12.

[13] Wong, Ruth. "Basis psychological needs of students in blended learning." *Interactive Learning Environments* 30, no. 6 (2022): 984-998.

Author's biography

Michael Dolinsky is currently working as a lecturer in Gomel State University named after Francisk Skorina from 1993. Since 1999, he is leading developer of an educational site for the University (dl.gsu.by). Since 1997 he is heading the preparation of the scholars in Gomel to participate in programming contests and Olympiads related to informatics. He acted as a deputy leader of the team of Belarus for IOI’2006, IOI’2007, IOI’2008 and IOI’2009. His Ph.D is dedicated to exploring and developing new tools for digital system design. His current research interest is in the field of Computer Science and Mathematics ranging from early stage to new information technologies.