STUDENTS LEARNING IMPROVEMENT USING UNIVERSAL VIRTUAL VERIFICATION PANEL


Slovak University of Technology Bratislava, Faculty of Informatics and Information Technologies (SLOVAKIA)
pistek@fiit.stuba.sk, jelemenska@fiit.stuba.sk, jurikovic@fiit.stuba.sk

Abstract

Attracting students’ interest is an issue that is being addressed in all courses. In general, students are more interested in doing practical exercises than learning theory and calculating or designing things on a piece of paper. In courses devoted to logic circuits design it is especially important that the students have the possibility to verify their designs and to experiment with various variations. In this paper we present a novel method for teaching students the logic circuits. Using universal virtual verification panel, the lesson can be more understandable. Several applications have been developed and one of the best will be presented here.

Some years ago the real logic gates panels were used for this purpose, where students could wire their Boolean functions by interconnecting the correct logic gates. These panels were limited in size, the logic gates had limited number of connection ports, and also the types of logic gates were limited to the selected complete set of gates. Nowadays, the virtual verification panels can replace the physical ones with much more advantages than just the space savings: various complete sets of logic gates can be available in one panel, the number of gates and their ports can be set up as needed etc.

In the paper, we present the possibility of verification panel virtualization so that it retains some of the useful restrictions. We keep a limited number of logic gates which are available to implement a circuit and the combinations of gates types are restricted to selected complete sets of logic gates. On the other hand some extensions have been implemented as well i.e. interconnection with Espresso, automated circuit verification, circuit truth table generation, and hierarchy support.

In addition to learning part of the software, we put emphasis on creating a testing interface to back-check students’ knowledge and gained practical skills. Teacher can create different types of tests with different restrictions. This is a great benefit for students to have the possibility to pass the test in the environment they are already familiar with. After the test has finished, the system automatically evaluates the submitted tests in order to facilitate the teachers work.

The communication between the teacher’s server and the students’ client applications is based on a proprietary communication protocol providing the distribution of tasks as well as automatic collection of results. This solution contributes to the effectiveness and safety of the testing process.

Keywords: improvement, skills-based assessment, verification panel, logic circuit.

1 INTRODUCTION

Some years ago the real logic gates panels were used for logic circuits design teaching. Students could wire their Boolean functions by interconnecting the correct logic gates. These panels were limited in size, the logic gates had limited number of connection ports, and also the types of logic gates were limited to the selected complete set of gates.

With extended use of Electronic Design Automation (EDA) tools the physical verification panels became outdated. However, for learning purpose it would be advantageous to have a tool with certain restrictions e.g. on gates types, number of ports etc. The virtual verification panels can replace the physical ones with much more advantages than just the space savings: various complete sets of logic gates can be available in one panel, the number of gates and their ports can be set up as needed etc. The application was developed based on our experiences from testing description languages [5] which is basically another form of implementation of Boolean functions. In the paper, we present the possibility of verification panel virtualization so that it retains some of the useful restrictions. We keep a
limited number of logic gates which are available to implement a circuit and the combinations of gates
types are restricted to selected complete sets of logic gates. These restrictions force the students to
be creative in the design and to try to reach the best space optimization using the minimal number of
logic gates. In addition to these limitations, some extensions have been implemented as well i.e.
interconnection based on disjunctive normal form, automated circuit verification, circuit truth table
generation, hierarchy support – a logic circuit can be wrapped into a new component further handled
as a logic gate.

The proposed tool improves the presentation possibilities in teaching. Teachers can send to students’
a description of a circuit from the initial circuit function specification up to the final circuit. Students
have the opportunity to interfere with each of the sent samples, according to their individual needs for
better understanding of the actual matter.

In addition to learning part of the software, we put emphasis on creating a testing interface to back-
check students’ knowledge and gained practical skills. Teacher can create different types of tests with
different restrictions. This is a great benefit for students to have the possibility to pass the test in the
environment they are already familiar with. After the test has finished, the system automatically
evaluates the submitted tests in order to facilitate the teachers work.

The communication between the teacher’s server and the students’ client applications is based on a
proprietary communication protocol providing the distribution of tasks as well as automatic collection of
results. This solution contributes to the effectiveness and safety of the testing process.

Universal virtual verification panel of logical circuits is an application for personal computers providing
functionality to design logical circuits and verify the rightness of their function. Virtual verification panel
is the successor to hardware verification panels. In the past, these panels were mainly used for
educational purposes. Given the continuing growth in the share of modern information technologies in
the field of education, it would be the idea of simulation of hardware at a computer program, which has
indisputable advantages especially as concerns the financial as well as the functional page.

Our goal is to create an application serving as an operational tool to simulate the behavior of the
designed logical circuits. Application also includes a simple way to define the user’s own logical
components identified by number of inputs, outputs and logical function [1][2][4].

However, the main benefit of this system is its component for distribution of tasks created by teacher
to the specified group of students. This function supports e-learning as the effective way of education.

2 APPLICATION DESIGN

Universal virtual verification panel is expected to be, as the title says, universal. This means that it
contains all the basic logical gates and complete sets of logical gates [2][3][4]. In the other way, our
application is universal to be used with any hardware and software platform supporting Java Virtual
Machine.

2.1 User interface

User interface in our application is clear and intuitive [1]. We take into account usage of Tablet PCs [6]
which are widely used in nowadays. It is divided into three parts. The first part is the main desktop
determined for design and verification of logical circuits. The second part is the side panel offering all
the necessary logical gates and other added components sorted in structured tree. The third part is
the top panel including action icons e.g. Save the circuit, New circuit, Load circuit or
Activate/deactivate verification process. There is also the standard menu bar above this panel with
standard items File, Edit, Features, Connection, Tests and Help.

2.2 Functionality

Our application provides possibility for creating logical circuit from logical gates and components [7].
These are placed in the side panel and can be added to the desktop using drag & drop technique [9].
Interconnections are made by drawing lines between designated components. Verification process
operates in real time; however it is possible to deactivate it, what can by useful in some special
situation. The application differentiates all connections according to driven logical value.

The created logical circuit can by whenever saved to binary file being ready for next use. Next
functionality provides possibility for adding the created circuit to the library of external logical circuits.
After this addition the new circuit is shown in separated section in side panel between other logical gates and this external logical gate is available for use in other logical circuits. This function represents easy way to define own logical gates or modules of advanced logical systems.

In addition, our application offers possibility to get logical expression representing designed logical circuit, what can be useful especially in process of back-checking or debugging.

Other functions like logic gates labeling or exporting the circuit into vector graphics (SVG) are implemented to make this solution more useful in process of publishing logic circuit schemes.

The main function we focused on is support for online-testing of students’ knowledge of logical circuits design using one of the predefined complete sets of logical gates. For detailed description of this function, see chapter 4 TESTING.

### 2.3 System architecture

Architecture of our application is divided into seven components ensuring all the functionality. These components are same in both student's and teacher's version, but there are some differences in the inner structure of the components.

The main component is **Graphical User Interface (GUI)** which is displaying the main window of the application including the desktop area, components panel and all the necessary controls.

**GUI** closely interacts with **Verificator** – component responsible for logic function verification. Implemented self-designed verification algorithm is based on recursive transition of logical value starting from the outputs of circuit. Three-state Boolean logic including the states of Low, High and High-Z is provided to simulate real conditions.

**Saver** and **Loader** components are used to save designed file into the binary file and to load it back when necessary. For this purpose they both use serialization technique [8][10].

**Sender** with integrated **Timer** contains all the functionality necessary to send tests and example tasks and samples to the students.

**Receiver** is responsible for receiving the tests from students. For detailed information about the functionality of these components see chapter Testing. Fig. 1 shows a brief data-flow diagram of our application.

![Data Flow Diagram](image)

*Fig. 1. Data flow diagram.*
2.4 System versions

Final system is implemented in two versions. The first version is created for students and the second one is created for teachers. Each version is fully utilized for designing and verification of logical circuit.

2.4.1 Student

In addition to the standard modules for design and verification of logical circuit this version contains also modules for tests receiving, tests development (this module is special modification of module for logical circuits design) and sending tests back to teacher for evaluation. The student graphical user interface (GUI) is given in Fig. 2.

![Fig. 2. Student’s interface – solving a task](image)

2.4.2 Teacher

This version contains also additives modules for creating tests, sending these tests to students and receiving tests (the main difference between this module in student's and teacher's version is that teacher is expected to receive more than one tests back) and tests evaluation. The teacher's graphical user interface is given in Fig. 3.

2.5 Communication

As the distribution of the tests as well as collection of the answers is based on network communication, it is necessary to define the network protocol. We decided to create our own protocol providing desired functionality.

Our protocol is based on client-server architecture although the data will be transferred through peer-to-peer network. We decided to use client-server architecture because the client will communicate only with one network entity (teacher's application). A peer-to-peer connection was chosen because these two applications are within one network segment with no server or another controlling device.
The protocol working on local area network is socket-based [11]. A socket is software that establishes bidirectional communication between a server program and one or more client programs. The socket associates the server program with a specific hardware port on the machine, where it runs, so any client program anywhere in the network with a socket associated with that same port can communicate with the server program.

As number of client applications running at the same time is expected, application uses the technique of multi-threading (division into multiple threads). The server creates a thread for each newly created communication channel with client. Without this technique, the client would block the entire server in their communication. In our case, the server application communicating with one client could continue to receive requests from the others.

Communication is established only between server and client, so the exchange of information between the client applications is not allowed. Each client application has a unique identifier which identifies it to the server. Unique ID is get from a random combination of numbers and name of the student. Every attempt to anonymous connection is ignored by the server.

2.5.1 Connection establishment

Calling takes place in two ways. Broadcast and then direct communication between server-client and client-server.

At the beginning, server sends the hello packet to identify itself through broadcast and waits for a response. Client program which has received this packet answers it. In response it sends its unique identifier (name of student) and waits for confirmation. If the answer arrives to the server, the address of client is saved and confirmation is sent. Hello packets are sent periodically to make it possible to discover the new clients connected later.

After creating a table of addresses on the server side and depositing the server address on the client side, server communicates directly with the client application. Likewise, the client application does not send data through broadcast, but directly to the server address.

3 STUDENT TESTING

Our product creates a virtual classroom [12], in which students are able to obtain study materials and test their knowledge. This process takes place in several steps.
3.1 Registration

Each student, who wishes to attend a course, has to register. The teacher has the opportunity to see all registered course participants. Registration is necessary for correct identification of student in order to his evaluation. Registration data including username and password are stored in the separate file. There can be as many files as necessary, i.e. one for each course, group etc. The teacher is able to select the file for the certain group.

3.2 Creating example tasks and tests

Course leader can create and edit tasks and tests for students of the course in the editor present only in the teacher application. When designing tasks, teacher has few options. He can either define the logic function while students have to design the circuit, or he can design an incomplete scheme and students are expected to complete it to provide required function. The teacher can limit the full set of logic which can be used to solve the example task or test.

3.3 Taking the tests and example tasks

Designed test or task is automatically sent to selected group of students forcing them to close all the other open schemes and starting the timer after student’s confirmation. After the time limit expires, tests are automatically sent back to teacher and stored as binary files being available to reopen for the purpose of back-check and assessment.

3.4 Test evaluation

To simplify the process of assessment, the simple tool for test evaluation is integrated. This tool generates the logic function of designed circuit and compares it with the function pre-defined by teacher. Based on the result, the status of test is changed to either Passed or Failed.

In the Fig. 2 and Fig. 3, there are shown brief diagrams of use-cases for both student and teacher.

4 CONCLUSIONS

We present our approach "Universal virtual verification panel of logical circuits” which will be a great benefit in exercises with the issue for the design and verification of the functionality of logic circuits. The program system has the same qualities as a physical verification panel and besides them we add a few useful properties. It serves as a good tool, whether in teaching or in a midterm tests. Application is useful in process of creating tests allowing teachers to define different limits for the specific test, such as default logic sets to create a logical circuit. The application also provides possibility for teacher and examiner to verify whether the students designed circuit correctly and to collect the test automatically after the specified period of time. Universal virtual verification panel of
logic circuits could be seen as an application for test creation and control, but it is not so. The application can be used as a tool for teaching and creating logic circuits, which will be verified in real time, or creating their own logic gates which will be saved and then re-opened and used. We have completed extensive testing of our approach and it will be used in the process of teaching of design of logical circuits on our faculty.

Fig. 5. Use-case diagram of actor Student

ACKNOWLEDGEMENT
This work was supported by Slovak Science Grant Agency (VEGA VG 1/0649/09 “Security and Reliability in Distributed Systems and Mobile Networks”).

REFERENCES


