

wiRed Panda

A Didactic Purpose Software for Simulating Digital Circuits

**Davi M. Morales, Lucas S. Lellis, Rodrigo T. Alves,
Lucas A. R. Pinheiro, Fábio A. M. Cappabianco**

¹Instituto de Ciência e Tecnologia – Universidade Federal de São Paulo (UNIFESP)
São José dos Campos, SP – Brasil

Abstract. *This paper presents a didactic software for editing and real-time simulation of digital circuits. The wiRed Panda is a open and free tool, developed to assist in the teaching of digital logic, also offering unprecedented functionality to generate code compatible with the Arduino . The software was used in digital logic classes with good approval ratings by the students, who considered the wiRed Panda as intuitive and practical.*

1. Metadata

Version described in the paper: 2.1

License: GPL 3.0

Link to source code repository: <https://github.com/GIBIS-UNIFESP/wiRedPanda/releases/tag/2.1>

Link to project website: <http://gibis-unifesp.github.io/wiRedPanda/>

Link to communication channel: wiredpandaunifesp@gmail.com

Programming languages utilized: C++

Dependencies: Qt 5.6+

Operating systems compatible: Ubuntu 12.04 or later, Windows 7 or later.

List of contributors: Davi Morales, Lucas Lellis, Rodrigo Torres, Lucas A. R. Pinheiro, André Leiniö, Daniel Renato, Guilherme Costa, Gustavo Oliveira, Héctor Castelli, Jimy Suenaga, Johnny Michael, Lucas Shinoda, Rodrigo Ramires, Thales Koba.

2. Introduction

The discipline of digital logics is one of the bases in the curriculum of professionals in the areas of informatics and information technologies. Given its importance, it is essential to offer courses of that discipline with the most innovative and practical resources, including digital circuit simulators.

The use of didactic tools, like digital circuit simulators, in the digital circuits class is important to give to the students the opportunity to use their creativity and improve their logic thinking, followed by challenges that will encourage their curiosity and provide a deeper understanding about the behavior of digital circuits.

Currently, there are a few softwares used for the implementation of digital circuits, each of them with its own features. Some of them such as Quartus II [Ciletti and Mano 2007] are proprietary, despite stable and reliable, require a huge amount of disk to be installed and lack clarity in the user interface, demanding a very high effort from beginners, and does not have support to real-time simulation of logic

circuits. Also, some of them, like Logicly[Tynjala], are stable, easy to use and powerful, but not free.

Finally, there are softwares like Atanua[Komppa], that are powerful and free, with full support to real-time simulation of complex circuits, but are highly unstable and lacks periodic support. From this perspective, we see that there are no fully appropriate programs for educational use and which would encourage enthusiasts to go deeper in the area.

We propose a novel multi platform software called wiRed Panda to fill this gap, which is a didactic and intuitive platform that can be successfully used to teach digital logic, allowing the implementation of virtually any digital circuit. The software is free and open source, available for any students, teachers, professors, or enthusiasts. wiRed Panda was developed in a relatively short period of time, with a reduced team composed of one professor and four students, small contributions and suggestions from the community, and with almost no financial support.

3. Frameworks Presentation

The software was implemented using Qt development platform¹. Qt offers support for the development of multiplatform graphical applications, with a variety of out of the box containers which follow well defined project patterns[Ezust and Ezust 2006] such as widgets, auto translation service, scenes and graphical items, among other essential tools. Qt counts with a very large world wide community, which is ready to help solving bugs and providing solutions by means of forums and open projects. QNodesEditor is one of such open projects, used for the development of wiRed Panda. It allows the creation of a graphical interface composed of nodes that may be linked by flexible wires².

Other free softwares like Inkscape were essential for the design of icons and images of the interface used by wiRed Panda³. The usage of simple, clean, and meaningful icons to access menu options and to represent logic circuit elements is among the key factors for the software to have a better usability. The visual identity adopted by wiRed Panda was chosen to offer an appealing experience to the users and, at the same time, to be practical during circuit development.

3.1. Software Architecture

The most important classes in this software are the ones related to the digital circuit representation. Each input (e.g. push button), output (e.g. led), logic gate, or memory (e.g. D flip-flop) is a graphic element, which has one or more input and output ports. One element can be linked to another, generating connections. Connections are always created between an input and an output port. These concepts are used for circuit implementation in the user interface and for signal propagation during its simulation.

¹Qt is a toolkit for the design of applications written in C++, and have an extensive graphical user interface framework. Available at <https://www.qt.io/>. Accessed: 2016-06-23.

²The source code and description of QNodesEditor is available at <http://alcoholic.eu/qnodeseditor-qt-nodesports-based-data-processing-flow-editor/>. Accessed: 2016-06-23.

³Inkscape is a professional vector graphics editor, available at <https://inkscape.org/pt/>. Accessed: 2016-06-23.

This relationship is described in Figure 1, in which the ‘GraphicElement’ class represents the elements (i.e. logic gates, inputs, outputs, and memory), the ‘QNEPort’ class represents their input and output ports, and the class ‘QNEConnection’ represents the wires or connections.

The ‘GraphicElement’ class is an abstract representation which is generalized as logic gates, inputs, outputs, and memories. Classes ‘And’, ‘Or’, ‘InputButton’, and ‘Led’ are samples of possible generalizations of ‘GraphicElement’, which contain a description of their respective behavior and appearance.

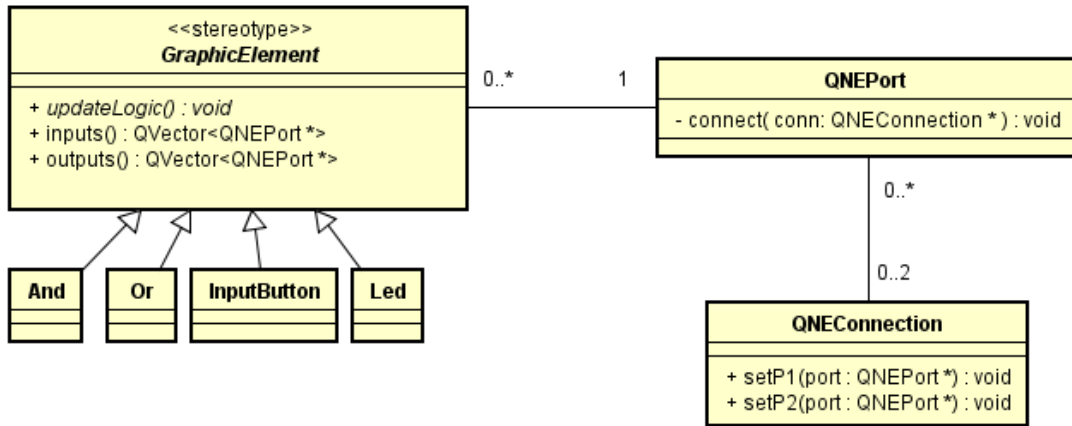


Figure 1. UML diagram of the relationship between the classes that compose the logic circuit’s representation.

3.2. Simulation Algorithm

The algorithm for the simulation of sequential and combinational circuits runs on a directed graph in which the vertices are logic gates and the arcs are wires connecting their input and output ports. There is a distinct logic for each gate which simulates its operation, that is, determining how signals should be propagated from its inputs to the outputs.

The computation of signal values follows a depth-first propagation strategy. As each node is reached, it is inserted into a forest (an acyclic graph that is not necessarily connected). According to [Bondy and Murty 2008], whenever a vertex inside a tree (or forest) is reached again in a depth-first search by means of a new edge, it means that we should not include this last edge into the forest, since it is a back-edge, that otherwise would produce a cycle. The vertices that do not reach other non-visited vertices during the propagation are marked as leaves.

Given the final forest, all gates are ordered from leaves to the roots, generating the order in which the signal propagation should be performed. Note that the same order is used to output Arduino code, which is discussed in Section 3.3.

As an example, the circuit described in Figure 2 is a D-Latch, a sequential circuit. This means that the status of the output of some logic gates are propagated back in the circuit and used in the next iteration. This same circuit is described by a graph representation in Figure 3, which is a directed graph with a loop between the vertices *f* and *g*. In this

- Inclusion of libraries.
- Declaration of constants, representing circuit's inputs and outputs.
- Declaration of global and auxiliary variables representing the logic gate's.
- Implementation of *setup* function, where inputs and outputs are mapped to I/O pins.
- Implementation of *loop* function. Given the input pin values, the signals are repeatedly propagated through the circuit of Arduino to the output pins, following the order defined by the algorithm described in Section 3.2.

4. Software Interface and Functionality

wiRed Panda is a real time simulator, and hence allows the users to build and edit the circuit while it is in execution. The used graphical interface is composed by the following elements:

- a tool bar for file and project management (e.g. save, load, new) and circuit edition (e.g. copy, paste, rotate);
- a search field to locate components among logic gates, memories, inputs, and others;
- a side bar, with four tabs containing: *inputs and outputs*, *logic gates*, *memory elements* and *sub-circuits*. These are elements that may be used to compose a circuit;
- a canvas, in which the circuit is implemented and simulated.

We implemented a drag and drop interface to add components from the side bar into the canvas. The wires are connected between gates by pulling wires from a gate port to another also in a drag and drop fashion. The input and output ports are represented by small gray and red squares, respectively.

Whenever a component is selected, a context menu appears in the side bar, containing all the attributes of the selected component. For instance, for an *and* gate one may choose the number of input ports. For a *led*, the color and name of the output may be selected. If more than one component is selected at the same time, only the attributes in common appear in the side bar allowing the edition of all of them at once. Figure 4 shows the main elements of the graphical interface and an example of component selection.

Table 1 contains all the components available in the current version of wiRed Panda.

In order to allow the creation of logic circuits in a organized and powerful way, wiRed Panda is capable of representing any circuit as box that may be loaded and replicated into other circuits. The boxes may be selected anytime from the sub-circuits tab after the first time the box is created. The circuit represented by the box may be accessed and edited at any time by double-clicking on any of its instances. Figure 5 contains an example of the usage of the box feature.

Another interesting feature of the wiRed Panda is its capability of replacing components with similar inputs and outputs. The user may select one or more components (e.g. 2 *or* and 2 *xor* gates) and replace them by a similar (e.g. 4 *nand* gates) without the need to remake the wire connections. This way the user will save project implementation time. Figure 6 shows how replacement can be done by just right clicking and selecting the component type that will replace current one(s).

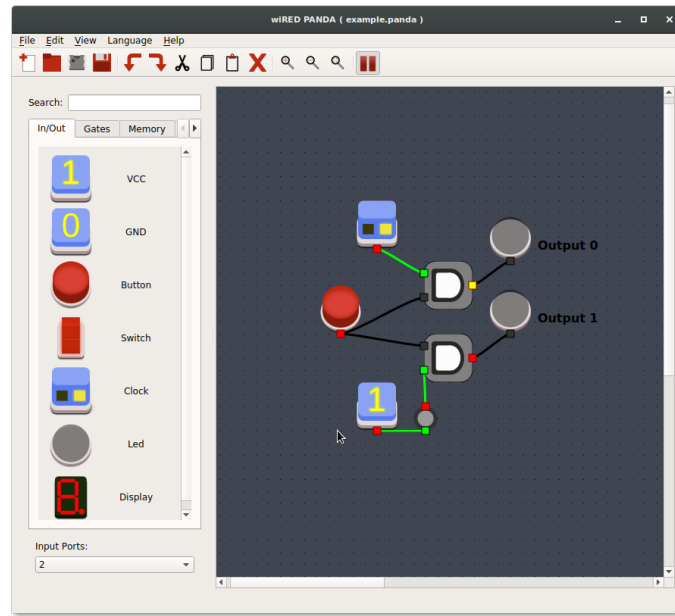


Figure 4. Plataform main features and an example of component selection.

Table 1. List of components available in wiRed Panda.

Input/Output	Logic Gates	Memory Elements
VCC	AND	D-Flipflop
GND	OR	D-Latch
Push button	NOT	JK-Flipflop
Switch	NAND	SR-Flipflop
Clock	NOR	T-Flipflop
LED	XOR	
7 segments display	XNOR	
	Multiplexer	
	Demultiplexer	

Finally, the proposed software also has other visualization tools such as zoom in and zoom out support, and the possibility to hide wires and logic gates for a cleaner layout.

5. Experiments

The first stable version of wiRed Panda has been adopted to develop projects in the discipline of Digital Circuits at the Federal University of São Paulo during the first semester of 2016. The feedback given by the students throughout the course was extremely important to improve the project and to define future works.

A survey was taken in order to evaluate the quality and efficiency of the software. Thirty volunteers answered it; they were all graduation students on Bachelor Degree in Science and Technology - with Computer Engineering, Computer Science and Biomedical Engineering trajectories - at the Federal University of São Paulo and were taking classes on Digital Circuits. They had experience with the *Altera Quartus II*[®] software at

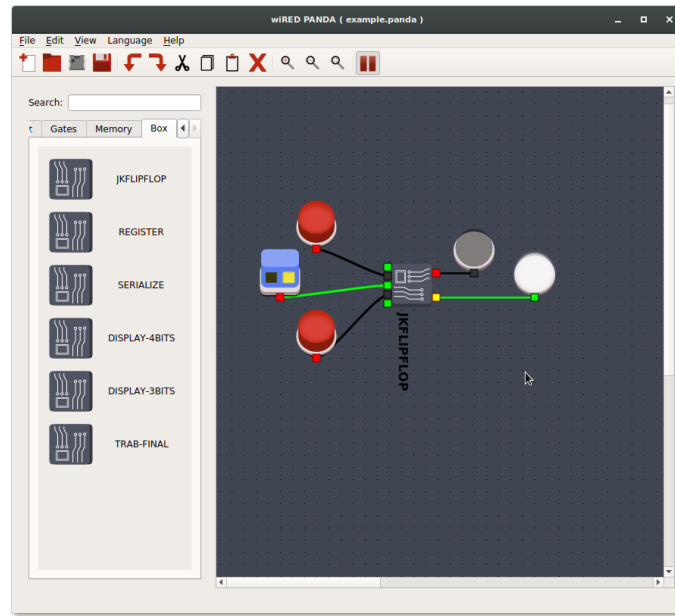


Figure 5. Example of circuit used as a box, and the sub-circuit tab.

the first half of the course and then with wiRed Panda at the second half. The survey was taken from 04/25/2016 to 05/01/2016.

The first question asked concerns to the overall quality of the software, presenting how cohesive it is at this point of the development process. At Figure 7 it's seen that wiRed Panda delivers quality in spite of an absence of bad reviews and a mode of reviews that attribute 80% of quality to the software, which is highly acceptable for such a young project and can be improved over time.

At Figure 8 there are the answers for a question that addresses the learnability aspects of wiRed Panda: "Was it easy to learn how to use wiRed Panda?". When it comes to a didactic software, learnability is seen as one of the most important - if not the most important - aspects of its implementation.

The most common answer to this question was 100%; also, the majority of the answers were positive. It means that the software is capable of teaching how to use itself in an intuitive and simple way. Unfortunately there was one bad review for this aspect of the software, which shows that it doesn't attend to all kinds of users yet and needs improvement.

It can be seen on Figure 9 that wiRed Panda delivers very high quality on usability; it matches the purpose of the software, which was meant to be lightweight, fast and to allow the users to edit their schematics just few clicks - or shortcuts - away from all the tools they need.

There was, like on the previously analyzed data, one bad review. Therefore, despite of its qualities, the software still lacks features that would attend all kinds of users.

Figure 10 presents mixed opinions about wiRed Panda's stability. The average value was 71.6% but with a standard deviation of 19.7%, which shows that wiRed Panda can be used on a daily basis by most people but still lacks stability.

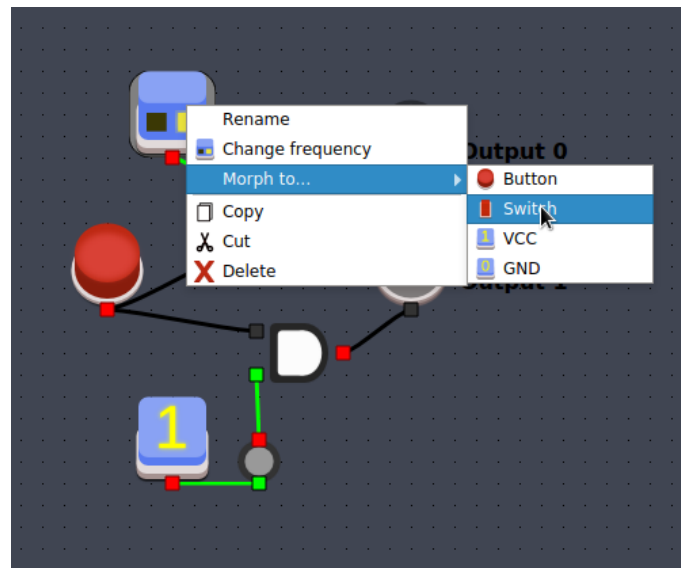


Figure 6. Execution of component replacement feature.

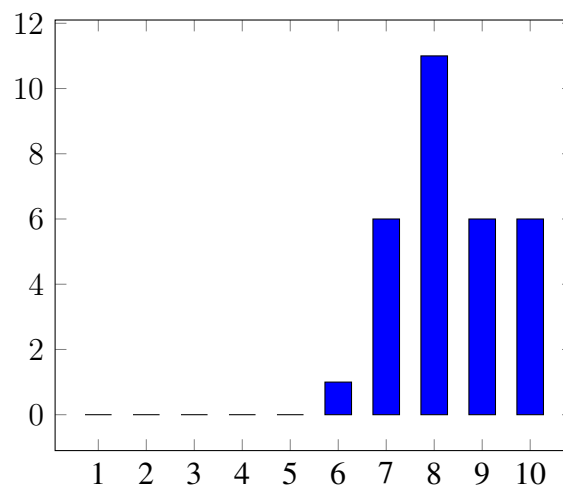


Figure 7. Answer to the question: "How do you evaluate the overall quality of wiRED Panda?". Higher values mean better quality.

We notice a general positive reaction from the students with respect to wiRED Panda. Analyzing Figures 8 and 9, it is clear that the main purposes of the software were achieved, making the learning process of digital circuits simpler and more pleasant. The time saved with the software can be directed to other academic activities. The practicality of the software allows more complex projects to be developed during the course schedule, deepening their understanding and experience with logic circuits.

Also, by means of Figures 7 and 10, it is clear that the stability of the software should be improved. This is probably the main factor to lower the overall quality of wiRED Panda. Nevertheless, the numbers are still reasonable and indicate one of the main directions in which the project should be improved.

Finally, in Figure 11, we notice that wiRED Panda has fulfilled the expectation of the students, since they would probably recommend it to others who want to learn digital

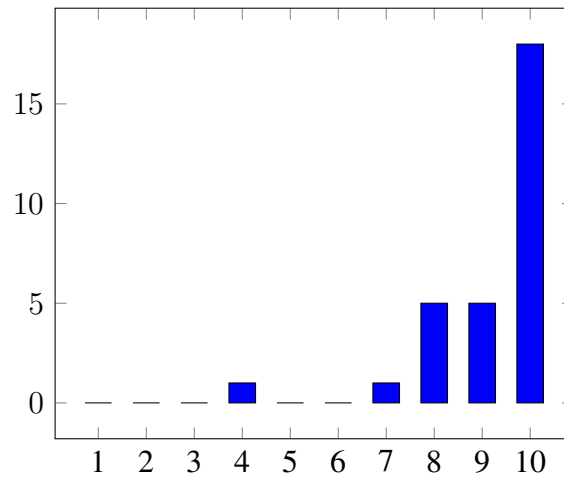


Figure 8. Answer to the question: "Was it easy to learn how to use wiRed Panda?". Higher values mean that the software is easier to learn.

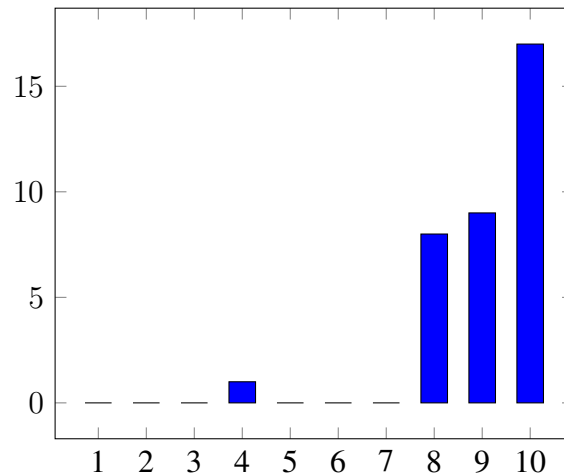


Figure 9. Answer to the question: "How practical is wiRed Panda in terms of usability?". Higher values mean greater practicality.

circuits rather than other softwares.

The validity of this project rests on its uniqueness in combining stability, quality, learnability and usability into a free and open source software. Therefore, wiRed Panda appears as a reliable choice for teachers, students and enthusiasts who want to learn about, design and test logic circuits.

6. Compilation and Installation Process

Currently, wiRed Panda is available for several operational system distributions. There are pre-compiled versions for Windows and Linux which may or may not require an installation process. In the case of the distributions that do not require installation, the user may simply execute a binary file to initialize the program.

If one wants to install the Windows version, there is a friendly interface which only requires the selection of the target folder. For Linux, the installation process is only available for Ubuntu 12.04 LTS or greater distros. In both cases, files with .panda extension

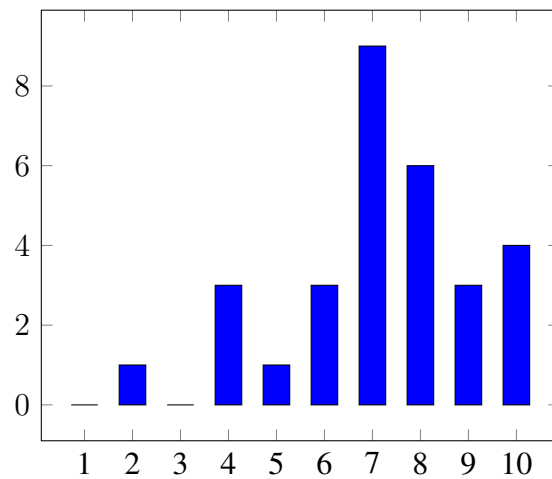


Figure 10. Answer to the question: "How stable is wiRed Panda to crashes and bugs?". Higher values mean more stability.

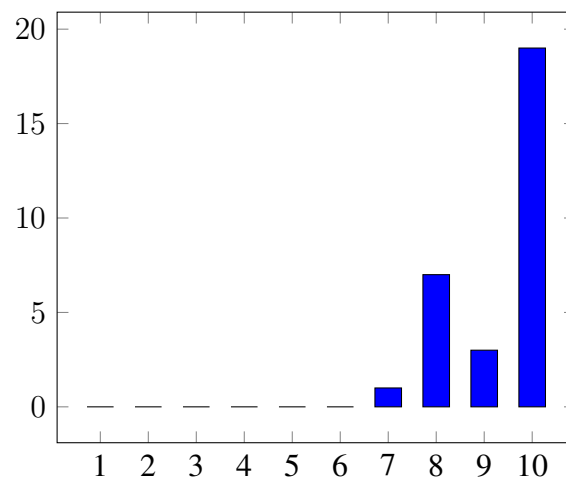


Figure 11. Answer to the question: "Would you recommend wiRed Panda rather than other similar softwares for learning digital circuits?". Higher values mean greater likelihood to recommend wiRed Panda.

are automatically associated with wiRed Panda by the operational system, being opened with a double left mouse click, and receive a specific icon that appear while exploring files.

There is also the most recent version of the software that can be downloaded from GitHub. In this case, compilation process is required, and it uses Qt 5.6 or greater versions. After the download, the wiRed Panda project can be loaded by QtCreator. QtCreator will configure the compilation settings automatically. Then, the project must be compiled and executed and the wiRed Panda window will pop up. After compilation, the binary file called 'wpanda' can be found in the build folder used by QtCreator. The software can be compiled by means of the 'qmake' command, followed by the 'make' command.

7. Conclusion and Future Works

The development and usage of wiRed Panda for learning digital logic by means of digital circuit simulation proved to be satisfactory in all intended aspects. It is a free and open source software which provides a nice and simple experience to users, with reduced learning time and reasonable stability. Also, wiRed Panda has the differential feature of Arduino code generation.

Future works include improvements to the performance of the project in dealing with sub-circuits, debugging, inclusion of more components (i.e. circuits, inputs and outputs), automatic hardware description language code generation such as Verilog and VHDL, and waveform simulation.

References

- [Bondy and Murty 2008] Bondy, J. and Murty, A. (2008). Graph theory, springer. Technical report, ISBN 978-1-84628-969-9. USR.
- [Ciletti and Mano 2007] Ciletti, M. D. and Mano, M. M. (2007). Digital design. *PHI*, pages 328–354.
- [Ezust and Ezust 2006] Ezust, A. and Ezust, P. (2006). *An Introduction to Design Patterns in C++ with Qt 4 (Bruce Perens Open Source)*. Prentice Hall PTR.
- [Komppa] Komppa, J. Atanua - real time logic simulator. <http://sol.gfxile.net/atanua/>. Accessed: 2016-06-23.
- [Smith 2011] Smith, A. G. (2011). Introduction to arduino.
- [Tynjala] Tynjala, J. Logicly - the digital logic simulator. <http://logic.ly/demo/>. Accessed: 2016-06-23.