Requirements and Analysis Document for Boolean Circuits

1 Introduction

1.1 Purpose of application

To introduce a new application with user-friendly features for creating boolean circuits.The application is primarily for educational purposes targeting students studying fundamentals of digital systems and computers or anyone else interested in the subject. It will allow the user to build circuits consisting of basic logic gates and simulate them in real time.

1.2 General characteristics of application

A stand-alone program with a sovereign posture though easy to use since it is a tool for educational purposes. The application will work on win/mac/linux platforms since it is developed in Java.

1.3 Scope of application

The user should be able to build and simulate complex circuits with the provided components in the application. However, the program’s scope does not include an algorithm for optimising circuits. It should also be possible to switch representation between US standard and IEC standard for the ease of use for anyone.

1.4 Objectives and success criteria of the project

To be able to build logical components and circuits based on elementary logic gates and with these it should be possible to create any, within reason such as memory-wise, complex circuit and have it show correct output from any given input.

1.5 Definitions, acronyms and abbreviations

The following expressions and abbreviations will be used in this report:

- *Logic gates* will be referred to as *gates*.
- *Workspace* is defined as the area in which the user place and connect components.
- *GUI* - Graphical user interface.
- *Java* - Platform independent programming language.
- *JRE* - Java Run time Environment. An additional software needed to run a Java application.
- *ALU* - Arithmetic Logical Unit

2 Proposed application

2.1 Overview

A user-friendly program to implement and simulate boolean circuits.

2.2 Functional requirements

1. Be able to build an arbitrary boolean circuit by connecting logic gates
2. Be able to save and open arbitrary boolean circuits
3. Use a previously saved circuit by importing it into an existing circuit
4. Be able to use keyboard shortcuts and mnemonics
5. Be able to use an internationally standardised layout
6. Be able to clock the circuit with clock-components

2.3 Non-functional requirements

2.3.1 Usability

A more user-friendly interface, with a sovereign posture. Tests with five expert-users will be performed to verify the usability of the program. It should be possible to switch between an American and an internationally standardised layout.

2.3.2 Reliability

It should be possible to save ongoing projects.

2.3.3 Performance

Response should be immediate. Maximum response time should not be noticed by the user at any given time.

2.3.4 Supportability

N/A

2.3.5 Implementation

To achieve platform independence, we will use JRE, thus users will need to have JRE installed.

2.3.6 Verification

There will be an automated test for individual components. Testing will also be performed by expert users.

2.3.7 Packaging and installation

The program will be packaged in a jar-file.

1. Unzip file. Zip-archive will include application jar-file, all needed resources and README-document
2. Start program. No installation procedure required for the application. JRE is required to run the program.

2.3.8 Legal

Save, undo, redo and open icons are all taken from VisualPharm under Linkware license. The icons representing functions such as cut, copy, paste, start clock, pause clock and save as component are taken from IconArchive by artist Fatcow Web Hosting under the CC attribution 3.0 license. Cross button is from the artist Yusuke Kamiyamane under the same license as previous artist.

2.4 Application models

2.4.1 Scenarios

See APPENDIX

2.4.2 Use case model

See APPENDIX

2.4.3 Static model

See APPENDIX

2.4.4 Dynamic model

See APPENDIX

2.4.5 User interface

See APPENDIX

2.5 Possible future directions

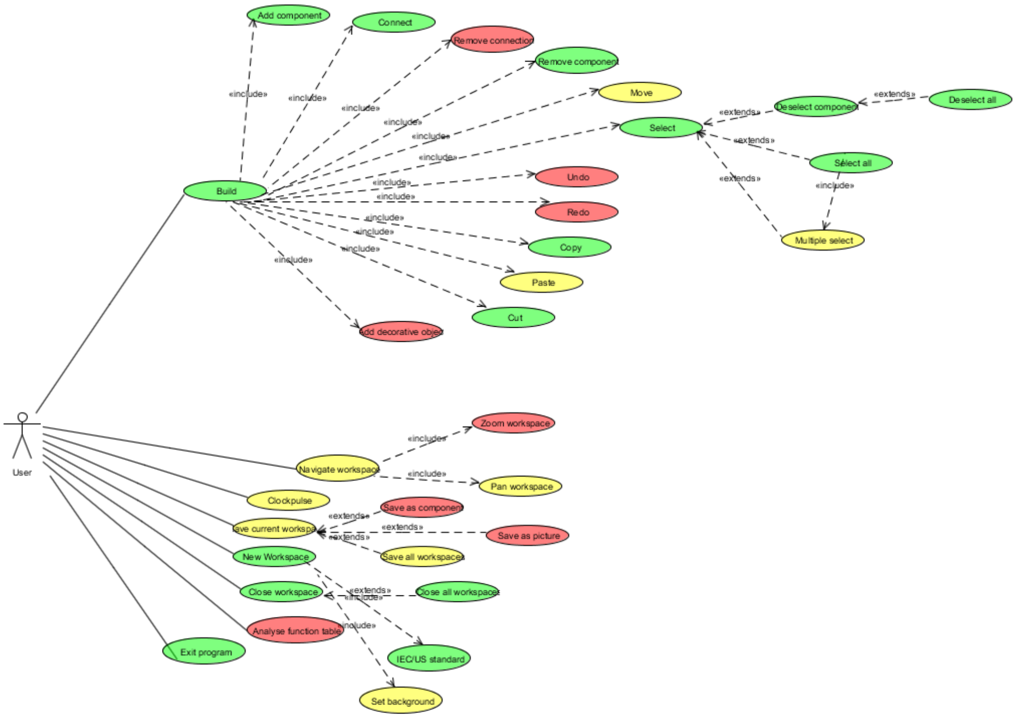
Assembler programming. Possibly more animations in a better GUI. Ability to export to known useful formats like .png. There would be convenient to be able to save a circuit as an individual component for later use. Functionality such as drawing Karnaugh maps, drag and drop functionality to place components and zoom workspace and other possible features that could be implemented in the future.

2.6 References

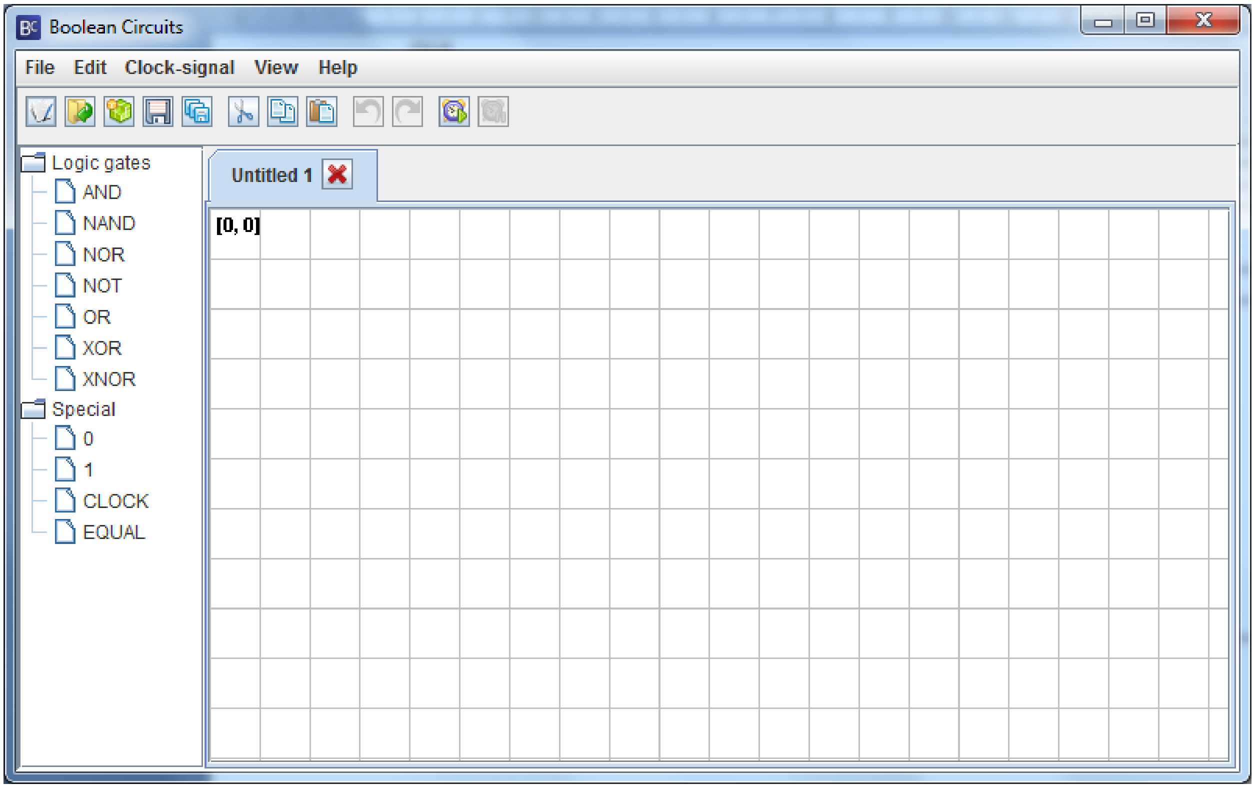
VisualPharm: http://www.visualpharm.com/
IconArchive: www.iconarchive.com/artist/fatcow.html

APPENDIX

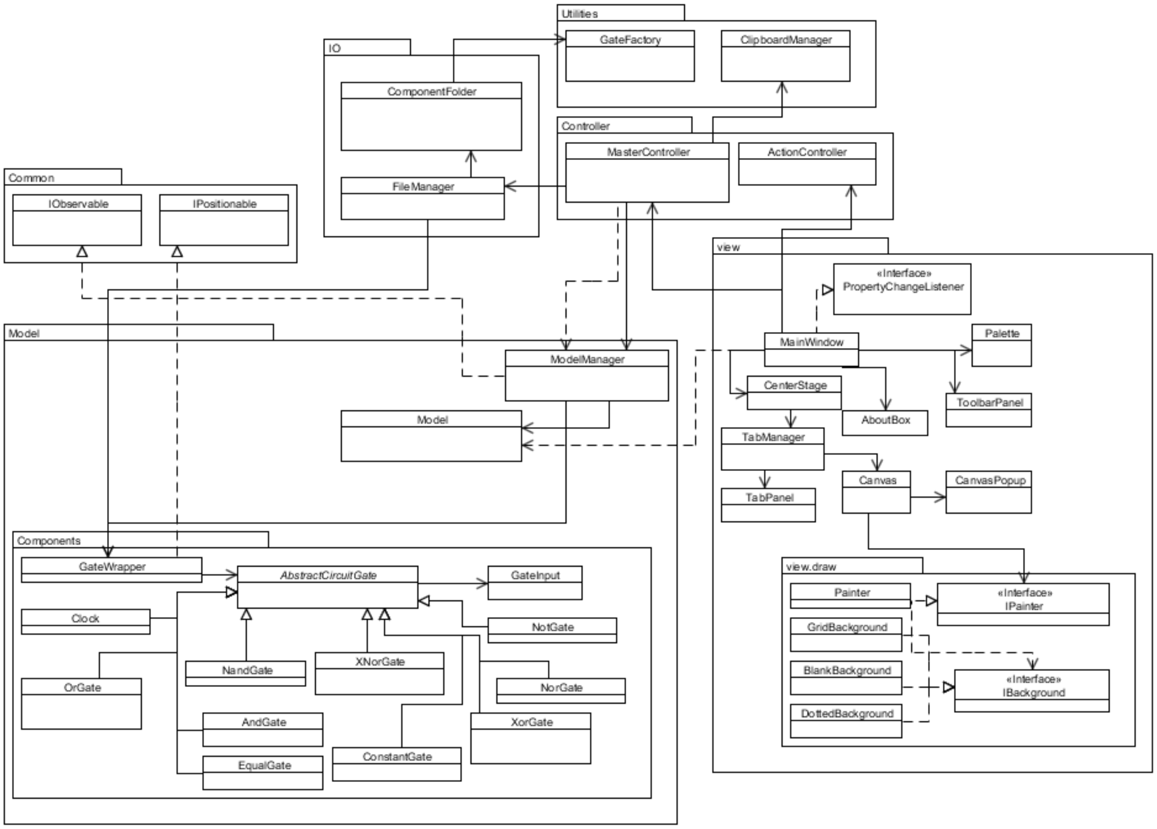
Use cases



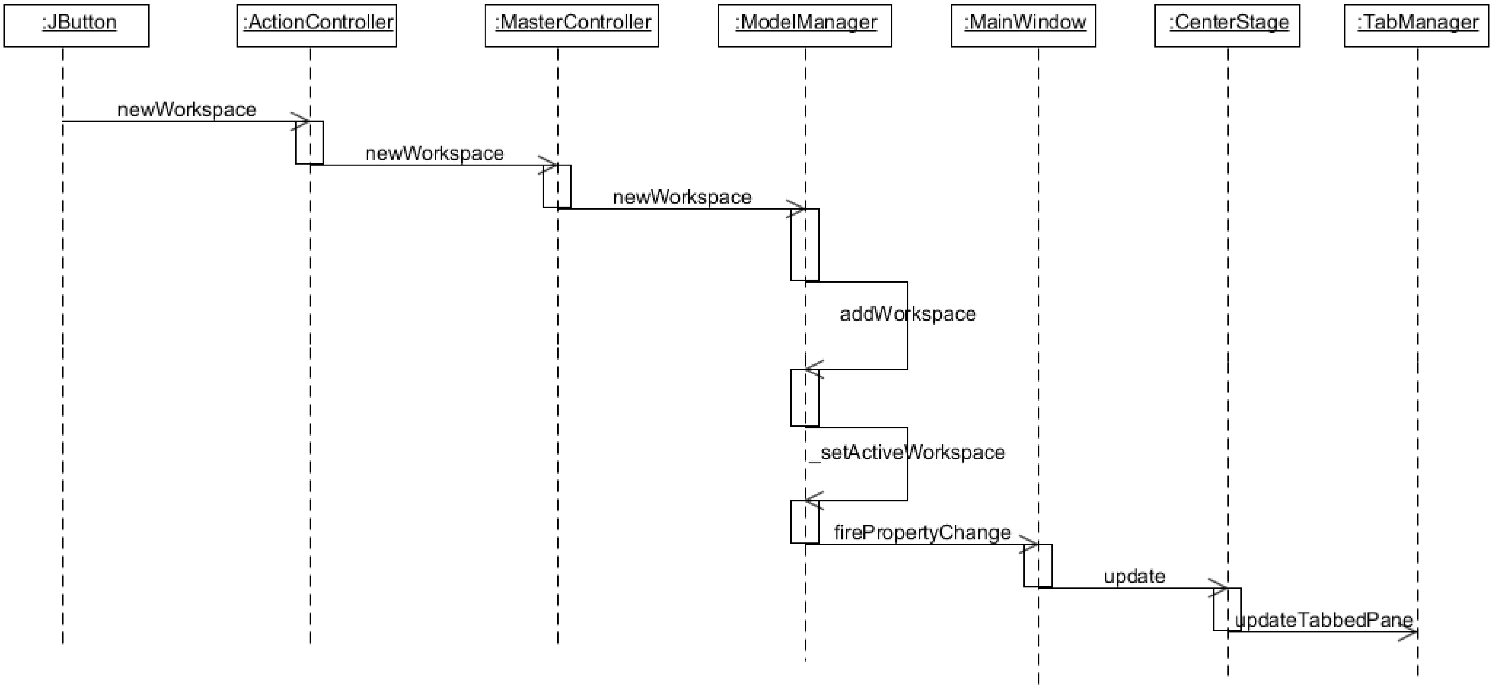
GUI


Boolean Circuits
File
Edit
Clock-signal
View
Help
Logic gates
AND
NAND
NOR
NOT
OR
XOR
XNOR
Special
0
1
CLOCK
EQUAL
Untitled 1
[0, 0]

Static model



Dynamic model


:JButton
:ActionController
:MasterController
:ModelManager
:MainWindow
:CenterStage
:TabManager
newWorkspace
newWorkspace
newWorkspace
addWorkspace
_setActiveWorkspace
firePropertyChange
update
updateTabbedPane