|  |
| --- |
| FONTYS 2013 |
| Digital Circuit Pro v1.0 |
| PCS5 Project |
|  |
| **Constantine Indjov, Cvetomir Yotov, Vladimir Mohamed & Zigmas Slusnys** |
| **10/24/2013** |



Contents

[Explanation of choices made 1](#_Toc370688302)

[Visual decisions 1](#_Toc370688303)

[Functional decisions 2](#_Toc370688304)

[Work division 2](#_Toc370688305)

[Problems encountered 3](#_Toc370688306)

[Boolean Value 3](#_Toc370688307)

[Overriding InputPos.X/Y 3](#_Toc370688308)

[Keeping all values updated 4](#_Toc370688309)

[Individual opinions 4](#_Toc370688310)

[Cvetomir 4](#_Toc370688311)

[Vladimir 4](#_Toc370688312)

[Konstantin 5](#_Toc370688313)

[Zigmas 5](#_Toc370688314)

# Explanation of choices made

First of all, we wanted to achieve simple interface and easy user interaction with it. This means creating or modifying anything is done within 1-2 clicks. In case of user errors, there is feedback for the user, so he knows what he did wrong.

## Visual decisions

* We are aiming to use easily recognizable images for the different objects.
* Red/Green colored connections according to their value
* Colorblind mode switches the Red & Green color to Blue & Yellow, because of possible users with daltonism.
* Each item has different input & output positions, where the connections are made.
* Any changes to values are immediately displayed visually.
* Sink & Source have two visual states, depending on their value.

## Functional decisions

* Whenever there is a change of a value, remove of connection or other kind of change in the circuit, the values are re-calculated for the whole circuit. That way we don’t have trigger for starting simulation, cause it’s always running.
* New Form – opens new form, which works individually with all the functionalities of the original one.
* Save/Load/Reset options, allowing the user to organize his amazing creations easily.
* Removing items also removes all the connections they have.
* Removing connections resets the value, given by it, in the receiving item.

# Work division

The work in our group was done together until the implementation, because we think that’s better way of making decisions for the problems we have.

**Code Implementation part**

During implementation of the code, we were working on one piece of code at the time, so most of the things are decided together.

There were a few parts of the code, where some of us focused more than the others:

**Cvetomir –** Save/load/reset options, also visual decisions

**Zigmas –** Connections

**Konstantin** – Connections

**Vladimir** – Connections

On the rest of the code we were working as a group.

# Problems encountered

## Boolean Value

First, we made the Values in Boolean type, but at some point we realized that it doesn’t offer us the needed functionality, because we cannot check unset value state, Boolean has to be either **true** or **false**, cannot be **null.**   
**Solution:**

Changed the **Boolean** to **Integer**, so now we use -1 as a **null** value, 0 for **false**, 1 for **true**.

## Overriding InputPos.X/Y

Every Item has InputPos field, which is the end **Point** for the input lines(connections) of it. Since Gates **AND** & **OR** have two inputs, they have two input points.

“InputPos.X = somevalue” doesn’t work in the constructor of GateAND/GateOR classes.

**Solution:**

We created new temporary point called “tempPos” which is used to get the position we want, and then set it to InputPos.

this.InputPos = tempPos;

## Keeping all values updated

In order to have live feedback, whenever a value is changed, we have to update all the values by calculating them and pass them by the connections.

**Solution:**

When we **setInputFromConnection(),** we **Calculate()** in the end, so that way we get a new value, if the inputs are changed.

**Calculate()** calls **passValue()**, so that way every time we Calculate the value of an Item, it gets passed by the output connections, if there are any.

**RemoveConnection** also calls **Calculate().**

# Individual opinions

## Cvetomir

The project made us thinking deeper, not only coding. Creating such

complex program gave us view of how the things are organized and that

everything should be discussed and figured as early as possible.

## Vladimir

I think that this project was a way more difficult from the projects we did so far. It clearly showed us how we can combine the visual part of a program with the action behind the scene. I think that you can give us instruction in what way to implement the functionalities in order our program to be more flexible and code clear and to use the best techniques in programing.

## Konstantin

I found this project more interesting, that it appeared in the beginning. There were many problems, that appeared when we started implementing the code and that was nice, cause it made us think of more complex logic, than we’ve used in the projects so far. I’m happy with our final version of the program, they might’ve been more things added to it, but I think that we have done is user friendly most of the times and functions properly. Think project for sure taught us how to organize our files better as a group and how decisions should be made with other people.

## Zigmas

The assignment was tough one, but our group did a pretty good job even though we had some difficulties at the beginning of the project. Everyone had to put their backs into it to find a solution to this project. I have learned that no matter how hard it gets, you should lose hope. There were a lot of differences in our group but we managed them out and worked side to side to develop what we got. In addition to that, we could have designed a better program which had more usability, but it's not how good the program is, but how much do you learn working in a group.