Coursework write up

Each title(underlined) will be on a separate page.

# Content page

# Analysis

This is aimed at educational purposes and to teach younger years the basics and understand of how logic gates and logic circuits work. For the computer science department, they wanted a more sophisticated system that would allow more advance logic circuits like flip flop. They also required basic systems like save and open previous files and a help guide on how to use the program for the students.

This project required a good understanding of how to use WPF C# as this was my programming choice. I’ve had to came familiar with the XAML and how the classes in WPF worked. Because of the requirements I had to research into multithreading. This was convivence due to C# have a good support for it OOP and multithreading.

On research a product like the website [logic.ly](https://logic.ly/demo) would provide a good learning platform for the school without the $59 price tag. It would also provide software that can be accessed without internet and requiring 30 students to search the web all to find the same website.

# Interview

I’ve interviewed my computer science teacher (Mr Stephenson) on what functions the program should hold. I asked class mates on the design to the GUI and how it should interact. This allowed me to model and develop based directly on their requests and fulfil their needs.

Mr Stephenson wanted a clear and easy to use simulator with a file tools to plan lessons beforehand.

Features:

* Visible changes to let the user know where the actions are being taken
* Change speed of the simulator
* File save and open features
* Real time calculations
* Zoom in and out features
* Drag and Drop
* Interactive canvas

# General Objectives

Achieve a fully working logic gate circuit simulator that will work flawlessly and fulfil the required needs of the departments. As a minimum it should incorporate all of the gates on the physics and computing specifications in addition to allowing multiple outputs from each gate.

It should be a responsive and lag free programme even on the worst of machines. The size of the screen shouldn’t be an issue and resizes to whatever aspect ratio.

No known bugs be present in the end product.

# Data sources & Destination

All data for the project that will be stored will be based on the class called Gate\_class. The list of object will be saved so that the user can load it back up later if required. I believe this doesn’t oppose a security threat as the data is not autonomous and only storing information on the formation and data values of the logic gates and how they’re placed. Everything else will be reconstructed so that the file size is a small as possible. The file type will be JSON as they’re perfect for storing class and are easy to read again.

# Programmed in C#

C# is a powerful and useful language, when it comes to UI’s WPF supports a wide range of projects that can be made. C# has a great support behind it and a lot of resources. This proved important as this was the first time using multithreading.

In WPF all UI elements are their own class, this allows you to inherit the class and create custom UI classes throughout your program. Because of this you can override existing methods that exist in the UI class. Giving you direct access to the event handlers that are fired by the UI.

C# for me is also a lot neater and better structured due to you have more control over other languages. This is a curse and a blessing as it means you know what’s precisely happening at the moment but you’re not helped through it and have to do it all yourself.

# Justification of chosen solution

# Pros cons of 2 other languages

It is a language I know well and suitable for the project due to it’s OOP capabilities. It also gives a lot of control to the programmer which was required and will be talked about more. Mr Stephenson also wanted file saves which is more compatible when it’s easy to convert classes to JSON. I programmed it inside of Visual Studio 2019 using windows presentation foundation.

# Limitations

Due to how my UI is a canvas and not a box grid I could not implement a successful path finding algorithm for the wires that connects the gates. This was seemed reasonable by the computer science department as a task like that is implausible to add due to the infinite possibilities and hard to extrapolate data. I’ve worked around this by adding different colours to the wires so that it’s easier for the eye to track.

# Method and sources

# Identifying a third party

Further research

Prototyping and critical path