Ctrl Alt Elite

**Updated** Report

11/16/18

**Updated** Description:

We are going to make a classic Arithmetic Logic Unit with a full datapath. An ALU is the core piece in microprocessors, using data and instruction inputs to perform operations. Our ALU will add two numbers, subtract two numbers, and shift numbers left and right.

It will be able to be used as a simple calculator, and the logic functions for AND, XOR, OR and NOT will be used. It will also have to take into account overflow and carry over errors. Our ALU will have a memory register of the last operation performed results, and will have a set of operations and modes.

The circuit will be used by entering a 8 bit minimum integer, and be used as a calculator with the set of logic gates attached. For example, the user will be able to calculate 8+4 or 6-1. Another set of operations it will be able to do is 9\*3 or 2\*9.

The ALU will be able to do these operation though logic gates/ functions, and will be easy to understand with the accompanying writeup and diagrams. It would be used for simple calculations, adding, subtracting, multiply and dividing.

**In our original code we wrote, the program did not operate the correct way we thought it would, so we redesigned it to make it more simplistic and efficient.** Overall, this ALU would be a great and efficient addition for a CPU to conduct the computation of typical math functions, typical logic functions, and typical errors.

**Updated** Member Tasks:

**Research - Jules, Luke**

~~Writeup~~ **System Design**- Skyler**, Luke**

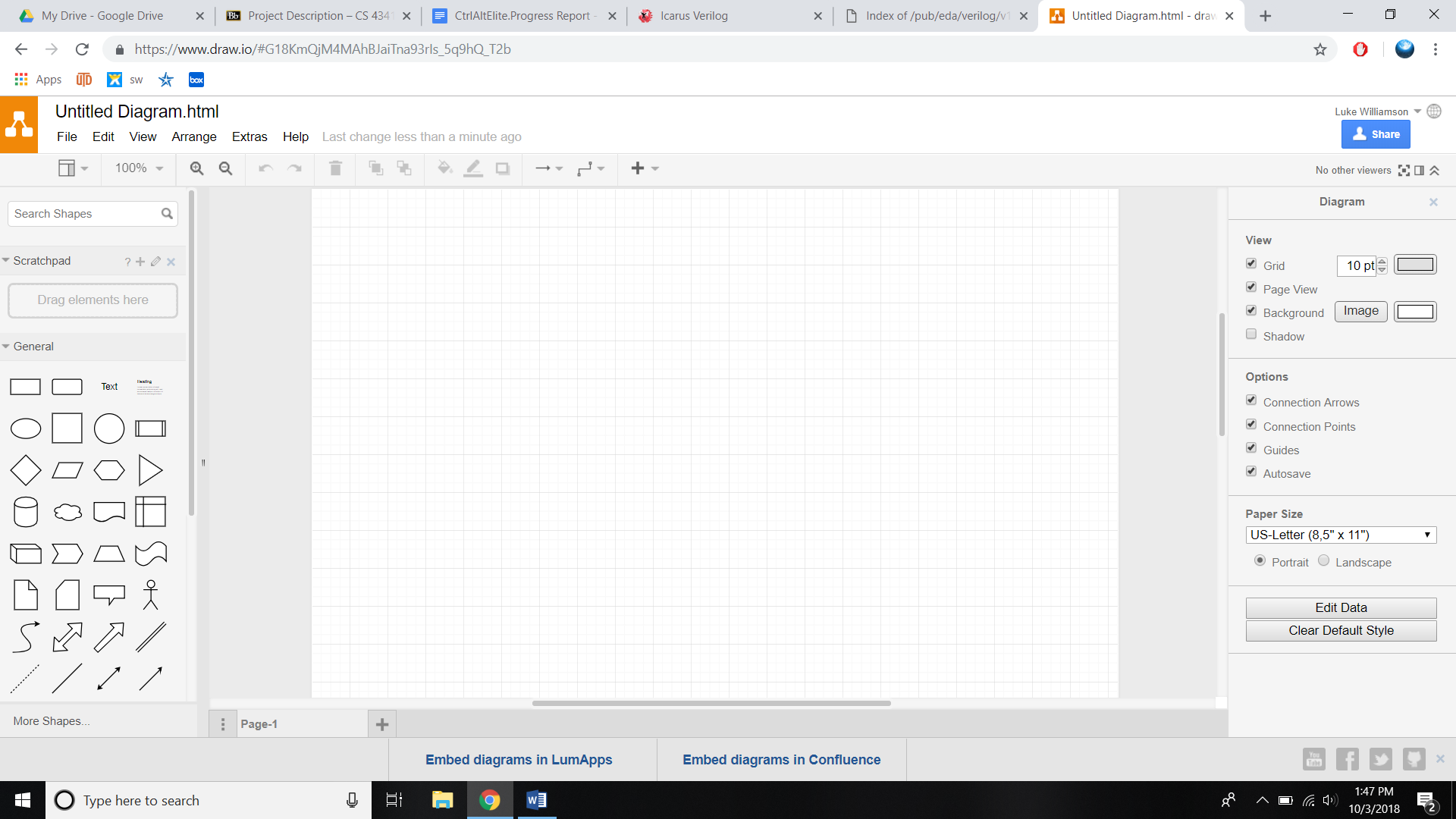
Diagrams- ~~Jules~~ **Skyler**

Code- Luke**, Jules**

Output- Luke/Jules~~/Skyler~~

**Updated** Software Discovery:

draw.io:



We chose to use draw.io for the diagram construction because it is user-friendly and has all the functionality that we require. This diagramming solution software was a perfect pick for our group since it is free to use with lots of affordable solutions. The software also has a vast amount of various symbols and tools to highlight the specifics for each diagram made. This makes it very intuitive for whoever will be looking at the diagrams made with this tool. There is also an option for customer support, which could be helpful for situations where there are errors within the diagram. Overall, this diagram solution software has a large amount of features that will make the diagrams easy to understand for our project. Furthermore, you can create professional-looking diagrams with it, thus they can be used in both academic and professional projects.

We are using Icarus Verilog (iVerilog) on codingground (from tutorialspoint.com) because it is easy to use and available online. **We also used the online verilog compiler, Jdoodle.** ~~The~~ **Both** compilers are readily available to use since it is accessible online for every cohort member to use. ~~This~~ **These** compilers are also very easy to interact with and very intuitive, so there have been no problems with the usage so far. **They** compile quickly and efficiently, and **they** gives detailed warnings. **Both** online compiler allows you to export your code to your computer, so that it can be shared between cohort members.

**JDoodle:**

****

**Updated** Participation Census:

In this pie chart, we are showing the distribution of each cohort member’s participation from the homeworks, cohort charter, cohort progress report, and cohort meetings. A detailed list is below of who has done what is below. Each task counts as one point.

* Homeworks:
  + HW1: Luke, Jules, Skyler
  + HW2: Luke, Jules, Skyler
  + **HW3: Luke, Jules, Skyler**
  + **HW4: Luke, Jules, Skyler**
* Cohort Charter: Luke, Jules, Skyler
* Cohort Progress Report: Luke, Jules, Skyler
* **Cohort Updated Report: Luke, Jules, Skyler**
* Cohort Meetings: Everyone has attended the meetings (we have had ~~2~~ **6** meetings outside of class.)
* **Project Meetings: Skyler, Luke, Jules (we have had 6 project meetings.)**

