# AC21009 Assignment 4 – The Manchester Baby Report

**Group:** 11

**Participants:** David Topping, Christian Zlatanov, Mathew Gallahcher, Stacy Onyango, Conor Kelly.

**Word Count:**

**Compile Command:**

**Run Command:**

## How we approached the assignment

Firstly, we talked on a Microsoft Teams chat and agreed as a group to research and have a full understanding of the tasks required before we met. During the initial call we discussed how we would manage the code; we decided on using GitHub to host our code. We then split our group into two, one half would work on the Manchester itself and the other half would work on the assembler.

For the Manchester baby we split things down into a few main categories: data structures, reading in the machine code, incrementing the control instruction register, fetching the program instructions, decoding the program instruction and executing the instruction. The aim was to try and split these tasks up as evenly as possible based on the time it would take to implement them and difficulty. This was achieved as well as we possibly could.

## Problems that we faced

When making the Manchester Baby Simulator, initially we didn't fully understand the brief. This meant when we went straight into programming the simulator we were making simple logic mistakes. We realised this was not a sensible way to face the problem so we stopped and planned each class with their functions and members. This worked a lot better and meant we had a clear understanding of the tasks.

It was initially assumed that everyone was confident using GitHub for code sharing. However, this was incorrect and let to mistakes with the GitHub repository that were very time consuming to fix. After that we made sure everyone was up to speed on how to use GitHub. Spending this extra time making sure everyone knew how to use GitHub saved us a lot of time overall.

It was a significant challenge working with binary numbers within the program and performing operations on the binary numbers. It took a lot of time to work out how to increment a binary numbers and also have them in a suitable form for our data structures. We used bitsets to help us deal with the binary numbers. A function was also found online that increments a bitset by one, this was extremely useful for incrementing the CI register.

## The solutions that you have created

We have created a Manchester Baby simulator. This has a main menu that allows you to run the default simulator. This reads in the machine code from a file, line by line and stores all of the data in a data structure called the store (an array of 32 registers). The CI is then incremented, the instruction is fetched from the store, this instruction is decoded and then executed. The main menu also allows you to run the simulator but with a custom store size, the custom store size allows you to have bigger programs etc.