**AC21009 Manchester Baby, Assignment 4**Group 9: *Lowrie Mouat, Matthew Gibson, Rachel Duncan, Timothy Baxter, Robert Kelly*  
Word Count: 769

**LINUX COMMAND LINE**

The exact Linux command line used to compile our program was “Ubuntu 22.04.1 LTS”.

**INTRODUCTION**

For this assignment, we approached creating a program that would simulate the Small Scale Experimental Machine (SSEN), better dubbed the “Manchester Baby”, written in either C or C++ (which is to be discussed as a project group and decided on). The Manchester Baby program was divided into two main components/goals; the hardware simulation of the Manchester Baby which would allow us to show the execution of machine code in binary being stored in computer memory, and secondly, the assembler which allows assembly language code to be translated into the binary machine code, which as stated previously would then be run through the hardware simulation of the Manchester Baby.

**MANCHESTER BABY DEVELOPMENT**

The development of the Cellular Automation program began by deciding as a group what language we would like to approach the project in (either C or C++) and most members of the group were overall impartial to the language that the program would be coded in so it was decided that we would write the program in C, rather than C++, due to the small preference that there was for the former language, due to more solid references and background in the language over the latter. After deciding the language, we would be using to code the Manchester Baby simulation program, we then split the development tasks into two (with half of the group working on the hardware simulation and the other half working on the assembler). As we are a group of six, as opposed to the regular five for this project, we decided on an even three-three group member split, however, one member of the group did not respond to the group chat formed on MS Teams, and therefore, we ended up with a three-two split with three members working on the hardware simulation and two members working on the assembler, and then a further delegation of documentation, however in practice the distribution of the work became skewed in development.

After assigning the basics of the work, a repository was created on GitHub to store and share the code, so that other members of the group could access it with ease. We then decided on how to store the binary characters within the simulation, it was suggested that there would be the option to assign them to unsigned 32-bit integers for the program memory efficiency, but it was then decided that overall, for the ease of programming that the binary integers would instead be stored within a 2D-Array.

For implementing the Manchester Baby, the test file provided was loaded to be run through the program (with a failsafe that if the test file was incorrect/not available to be loaded then the program would exit and bring the user back to the menu). An issue that we ran into in the hardware simulation of the Manchester Baby was that the CI was unable to count the line numbers correctly, stalling the program and making it unable to run through the rest of the binary code conversion as it wouldn’t be able to move past a line of code, effectively making the code be stuck in one place, this, however, was run through and adjusted so that the program runs through and converts all of the lines that are given. The code that is able to simulate the Manchester Baby runs through adequately but still has bugs and titbits within it that could be adjusted and fixed to allow the program to run more robustly and reliably, and ideally, if we were able, we would have liked to implement more functionality.

Overall, disregarding the group member that was unresponsive, communication within the group was clear and concise, with group members contributing to the group chat conversations and the work itself, with their assigned tasks, and members were receptive to questions and suggestions that were given by other group members. A caveat and issue to bring up would be that development was started relatively late to the deadline due to group members’ commitments with other projects but these issues were communicated to the group at the time if something was impeding progress on the project, so the issues were mediated to the best of our abilities. And although the work was distributed initially, the distribution of the code to each member became rather disjointed with some members contributing more than others, though everyone was still communicative outside of the practical efforts.