**Manchester Baby Report for Group 2**

**Kelly Evans, Naomi Hemingway, Douglas Inglis, Daniel Niven, Mohammed Yakub**

**Word Count: 692**

**Compile Line: make**

**NOTE:**

If you wish to run the included ArrayTest-MC.txt, NchooseR-MC.txt, LinkedList-MC.txt or Cellular-Automaton-MC.txt (note for this file you may wish to build using make DEBUG=1 . this removes colours and makes the output fast enough to run the program in a sane amount of time, though the program will work without this, albeit very slowly due to terminal colours slowing down the output to the terminal) you must increase the memory, enable the immediate addressing mode and enable the extended instructions. To do this add the switches -e -i -m 128. Similarly, to assemble these files run the assembler with the added switches -i -m (alongside the arguments for loading the file as normal).

As soon as the teams for the Manchester Baby assignment were assigned, Douglas added everyone into a Teams chat and we decided on which platform would be the best for us to communicate in. We debated between Discord and WhatsApp as none of us checked Teams on a regular basis and since more people had a Discord account, we moved our main discussion to there. During our first in person meeting we talked about the project amongst each other to assess where our interests and knowledge lay. We figured that the work would be better if there was already an interest in the task, otherwise we might put it off till the last second and or do subpar work due to lack of caring. Douglas has already started a bit on the simulator as it was a coding topic he had found really interesting so we assigned him as the leader of that half with Mohammed and Kelly who both had no strong preference either way. For the assembler, we assigned Daniel and Naomi as they both had a rough understanding of where to start there as opposed to the simulator where they were both kind of lost on. Before we got really into coding, Mohammed created a word document for us to share problems and solutions that came up during coding. This simple step saved a lot of time when writing the report as well as allowing us to create a more detailed report at the end. We also took the time to learn about the Manchester Baby, with some people learning best by having another group member explaining what is going on and others finding an in depth explanation on YouTube that went more into the history of it all which wasn’t necessarily needed for the assignment but gives a reason why learning how this code works is important.

A major problem was making sure that the program ran on the QMB computers after we had started coding it. Thankfully we remembered to check before handing the code in because it wouldn’t compile there when it would compile on a personal computer. All we needed to do was a quick variable name change from FUNCTION to func and the program ran smoothly. Initially we had made plans to make a GUI that emulated the Manchester Baby, not just as options but as a window with buttons like in the original that a user could switch on and off and then see the output in a text box that emulated the screen in the original. This was going to be made using the GTK extension for C as we had chosen to program in C instead of C++ because we were all more familiar with C. However, in order to use GTK we needed to check if it was installed onto the QMB computers. Due to technical issues with the QMB we were only able to check this in the last week, and it turned out that they could not compile GTK library code so we had to scrap that idea much to our disappointment. A problem with a simple solution that had us hung up for a bit was in or div, divide, function. Because you can’t divide by 0 there had to be a check to make sure the operand was not 0, otherwise it would cause an error in the code. Despite the loss of our in depth GUI, we were able to add a bunch of extra instructions to our Manchester Baby simulator. So much so, that we had to increase our operands from 5 bits to 8 bits because we have created over 32 instructions. Delving into making a simple (by today’s standards) computer such as the Manchester Baby really opens your eyes to the complexity that modern computers must have if you have to expand memory for more than 32 instructions. Along with being insanely more powerful, the computers today are so much smaller than the Manchester Baby was. It is a bit awe inspiring that what we take for granted in our pockets today was the stuff of wild science fiction 80 years ago.