**The Manchester Baby**

**Team**

Sam Glendenning, Dylan Keningale, Will Lockett, Matthew Daldry

**Language used for development**

C++

**Compiler used**

g++

**Introduction**

Our task was to build a replica of the Manchester Small-Scale Experimental Machine (or Manchester Baby). This implementation was to replicate as many features as possible that the original from 1948, including that of its LSB binary, comparatively small amount of data storage and its small instruction set.

**Design**

As we were essentially building a computer, designing it was possibly the most important step. Indeed, about as many hours were spent discussing the design of Baby as were spent implementing it. Baby’s memory (or store) is made up of 32 lines of 32 bits of data, plus three registers used for navigating its store, decoding instructions and performing calculations. In our implementation, Baby’s store is represented by a two-dimensional array of Boolean values, 1 or 0, with options for between 32 and 256 lines. Each line is read from the store and stored in a string value, so that the binary can be interpreted properly. The operand is fetched from the first 5 digits and the opcode from index 13-17. Baby’s default instruction set allows for 7 possible instructions from 8 possible values (SUB instruction has two mapped to it). However, when we extended the instruction set, this required the opcode to be 4 bits long. The new instructions are ADD, for adding numbers, MUL for multiplying numbers, DIV for dividing numbers, MVF for copying the contents of the current instruction register to our new one and MVT for the opposite.

**Problems**

One major problem we encountered was understanding the design of the Manchester Baby. For the team, it was difficult to grasp how it functioned at first. This was probably one of the greater problems as it prevented us from accomplishing anything for a period of time.

Another issue the team encountered was a strange error where a certain value wouldn’t print to the terminal. The cause was never clear, as the code for this print was identical to another line exactly like it that did work.

Another problem the team faced was a misunderstanding as to how CI and PI were supposed to be implemented. As the team misunderstood their function, this lead to issues in the program later on in development.

**Solutions**

The team resolved the initial problem of grasping the concepts of the Manchester Baby after sitting down as a group and reading about it online until we understood. We read about it and asked the lecturer or tutors for help as well, spending plenty of time to ensure we all fully understood the project.

The next problem was solved by rewriting the code from scratch so as to remove the possibility of phantom typos. This did in fact correct the issue.

The next problem was solved as the team realised that out understanding of CI and PI did not fit with the project, causing us to take another look at their purposes. Upon realising our incorrect implementation, we set about correcting all code that involved the use of CI and PI.