Eli Granade's Journal

MILESTONE2:

10/4/23

I helped plan the next milestone, and finish up one. We decided to change our command conventions, opting for the final register to be the destination in all cases. In addition, to make typing faster, we decided to change the names to be mathematical symbols or sets of relevant symbols, such as Y for the branch command since it delineates a fork.

10/10/2023

Spent about twenty minutes contributing to the RTL, and considering what else needed to be done for milestone two, and future plans. Discussed with the team why the destination register was named 3rd arg. While we left it for the speed of jump commands, it may become an issue if we develop the processor to be a pipeline. We left it as is for now. I wonder if pipelining could be even faster, should we decide to disregard the actual final cost of the machine. One possible way to do this is by swapping all non-essential registers (i.e. the input/ return argument registers) to be saved or all to temporary registers. This may make the procedure calls better overall. Will bring up at tomorrow's meeting.

10/11

Proposed again the mark command. Was again refused due to the simplicity of the current commands. Decided to swap our work from multi-cycle to a pipeline system. It will be harder, but fastest possible system. Decided to double check the RTL by
Went over the work we did for milestone 2, and confirmed that it met the requirements.

For the next milestone, I will be assisting Drew on the datapath construction, probably taking less than an hour, and possibly taking some time to contribute to the tests and basic quartus work, depending on how long I have.

MILESTONE 1:

9/27/2023

We only spent the remaining class time that we had at the time.
We decided to do a load store implementation becuase it is the fastest and most simplest for our purposes.
We decided to have the following registers:
A zero/ground register, a return address register, a stack pointer, two argument/return registers, and three temporary registers. We chose 8 registers so our main add command can be 12 bits for registers and a four bit opcode. We chose a four bit opcode since we're restricted to 16 bit commands.

10/3/23

I helped complete the design document that is due for milestone one, spending an hour after class.
Gave us the ability to add more R types by extending the opcode of the R type instruction, while simultaneously removing dead bits in the instruction type.