Garrett Doolittle Work Log:

Milestone 1:

Sunday, October 1, 2023

Met with everyone to finalize processor type and create the design document. We decided on a mix between accumulator and mem to mem. Then later decided we're doing entirely memory to memory. We rewrote our instructions with memory to memory in mind. We started on the design document, with my specific roles being the abstract, the logical and verbose descriptions in the ISA table, and setting up the Github. [3 hours]

Monday, October 2, 2023

Individually worked on the code snippets to improve the code and translate it all to machine code. I changed the instruction type of mov to L because it uses a different format than the other instructions. [1 hour]

Tuesday, October 3, 2023

Met with the team made last minute changes to make sure we had everything needed for milestone one. [30 mins]
We also discussed what tasked we would do
I have 2 types of instructions for the RTL and 1/4 of the base component chart. [will take around 2.5 hours]

Milestone 2:

Wednesday, October 4, 2023
We met as a team and discussed milestone 2 in further depth and split up the work between the members to get work done over break. I also reworked some of the code snippets and added in the GCD snippet because I didn't realize we needed it in milestone one. 🙁 [2 hours]

Saturday, October 7, 2023
Worked individually on the RTL for SP. While I was working I realized that we had some typing errors for some of our instructions and fixed those along with adding in the JA instruction type. I also made the RTL for JA. [2 hours]

Monday, October 8, 2023
We met as a team to discuss what we worked on over break and went through our RTL's for errors and to make sure they all had the same two starting cycles so that everything wouldn't break. [2 hours]

Tuesday, October 10, 2023
Worked individually on our code snippets that I reworked from the last milestone. I added any missing comments to snippets I changed and commented on all the new snippets I added. [30 mins]

Wednesday, October 11, 2023
We met as a team and worked through the last few things of milestone two which included writing manual tests for the RTL and checking them along with fixing any errors that arose. [45 mins]

For the next milestone I will be working on writing the descriptions for our unit tests of the components. Along with writing and testing some of the components. I expect it to take 5 hours.

Milestone 3:

Monday, October 16, 2023
Worked individually to come up with testing specifications for the register and the alu. I was unsure of how to implement the mem and therefore the tests so I did not write a specification for those. [1 hour]

Later we met as a team and worked all pulling all of the stuff we did individually together and also discussed it all. We also came up with a way to implement memory and decided how to test it as a group. We then divided up the rest of the work. [1.5 hours].

Wednesday, October 18, 2023
Worked individually to write the test cases for the 8 bit and 16 bit registers. I also fixed some errors in the memory and alu that i wrote so that they could compile properly. [2 hours].

Next milestone I will implement stage 1 of the integration plan. [2 hours]

Milestone 4:

Monday, October 23, 2023
Worked individually to implement Stage 1 of our integration plan with some simple tests. I got stage 1 working for the most part. [2 hours]

I then met with the team and completed stage 1 then looked over the FSM and made sure that I couldn't find any major errors. I also worked with Ethan Hutton to implement stage 3. In this meeting we made a few design decisions like taking the clock out of the ALU and we added in the write flags in the registers. [2 hours]

Tuesday, October 24, 2023
Worked with the team and discussed what we had done and who was going to finish up what. I am going to write more tests for stage 1 and 2. [30 mins]

Next milestone I will work on implementing Stages 4 and 5 as well as writing more rigorous tests for all the earlier stages. [4 hours]

Milestone 5:

Sunday, October 29, 2023
I implemented Stage 4 and found that all the previous stages need more inputs and outputs to properly connect them in stage 4. I then started testing stage 4 and ran into multiple roadblocks. Mainly revolving around memory. I then went and worked with Wesley for a little bit and he showed me how to better implement memory based on lab 7. After this I worked a little more on it and got stage 4 to run a memory file and it eventually after lots of debugging ran a few different types of a and ai type instructions. [8 hours]

Monday, October 30, 2023
I was still coming off the high from Sunday of finally getting the process to run something and I started implementing stages 5 and 6 which were much simpler than stage 4 because of all the precursor work I had done in stage 4 to make them connect easily. I didn't really feel like running any test cases for stage 5 and 6 because the logic in them was very simple so I waited until Wesley and Ethan had finished the control. Once they did I implemented stage 7 which just combined stage 6 and the control and decided to run a simple A type instruction since I knew they had worked previously in stage 4. They did not work initially in stage 7 but after looking over stage 6 for a while I decided to write one A instruction test for stage 6 to make sure that it still worked after adding on the stuff for stages 5 and 6. It did.

Then I went back to the control and found that it was not changing the state properly and it wasn't resetting the variables from the previous cycle so it was changing things that were not supposed to change. I worked with the rest of my group to fix this problem and we got an A instruction to work. Then I thought screw it rel prime time since I couldn't see any big reason why it wouldn't work. So I typed out the code into a new relprime.txt and ran it. It crashed and

burned. At first it was just small stuff like certain muxes being set to the wrong value in the control. But then I ran into a big problem: the control was not changing to the right state out of state 2. After about 2 hours of debugging and not paying attention to Proofs at all, I figured out a solution. Keep the signals coming out of control updating on the positive edge but change the actual control state on the negative edge. This fixed the big issue with running rel prime and then after running that I realized there was also a problem with the code we had written so I fixed that and rel prime ran successfully. [8 hours]

Tuesday, October 31, 2023
I was happy with the processor but I am also loving this project so I had to keep going. Spent way too much time getting started on the special feature. :) [5 hours]

Wednesday, November 1, 2023
Worked with the team to work out some kinks in our processor like not having in/out and also our memory making it take forever to compile in quartus. I then remade the memory but better and it made our processor run about twice as fast but it only made it compile slightly faster. [2 hours]

Next milestone I will work with my group to put finishing touches on the processor and calculate all the stuff. [est. 5 hours]

Milestone 6:

Thursday, November 2, 2023 - Monday, November 6, 2023
Worked on the minecraft implementation of our processor. [25 hours]

Friday, November 3, 2023
Implemented a new form of relprime that runs substantially faster and also added in Memory mapped IO. IO gave me quite a bit of trouble but most of it was because I wasn't using verilog correctly. [5 hours]

The End:

Thursday, November 9, 2023
Fixed our issue with memory and got the processor working even faster. It now compiles as actual memory and reads/writes on the not positive edge of the clock. [2 hours]

Friday, November 10, 2023
Finished implementing I/O in the minecraft processor. Started work on the presentation and the final doc. [2 hours]

Sunday, November 12, 2023

Finished the presentation, practiced and finished the final doc. It is Done. [5 hours]