NAME? Title?

Log begin 10-01-2023:
The current date is October first twenty twenty three. The project group met for a total time of 3 hours to discuss the implementation of the processor. The meeting started with debate over the chosen architecture. The original architecture was a load-store type that would model other load-store architecture in its implementation. The process of the group went as follows:
the professor will grade us more strictly if we do load-store, we need to do another type of processor. The options are an accumulator or memory to memory type, the stack type was not considered for concerns for the sanity of the group. For a time the idea of a hybrid memory to memory accumulator was discussed. This idea was scrapped in favor of only using memory to memory because memory to memory was faster/more efficient than the accumulator under most metrics.
The final decision for processor architecture was a memory to memory based system.

I was responsible for a large portion of the debate over what architecture to design. I also created the current opcode schema and list of commands that are currently possible with the architecture.
Our group plans to meet on Wednesday to discuss assignment of work for the next milestone.

Of important note for our current design position is that there are really only three types of instructions, an ‘arithmetic’ type called a, a ‘immediate arithmetic’ type ai, and an argumentless stack operation. There is also a one argument jump, but that is not particularly relevant right now. The stack currently only stores the values of return addresses and the only operations that access it as of now are the ‘push’ and ‘pop’ commands that also move the PC to the value that was on the stack. The current plan is to have a specific place in memory that keeps track of the value of the stack pointer. → can just use a register!

Log Begin 10-3-2023:

I have split the stack access into two types: access that knows the PC and access that does not. I chose to do it in this manner because I am unsure what address we would hand the process if we were wanting to push the program counter, because as far as I know, it does not exist in memory.

lets discuss in our M1 meeting!