Stony Lake Mid-project report

(a) Progress to date:

- ALU: Complete. Implemented 10 operations plus rotations and tested. May need to adjust conditions for overflow/underflow. May need to change during integration.
- Instruction Decode: Decode framework and logic for 35 instructions is complete. Need to create documentation and determine control values for jump instructions.
- Control Signals: Designed and implemented a set of control signals to indicate what operations to complete during each T-state.
- Input Output (I/O): Complete; needs to be integrated and tested.
- Memory (index registers, program counter, data + program memory, instruction register): Complete; needs to be integrated and tested.
- Makefile: Complete as far as we can see, may need to be updated along the way.
- We have also written a function for reading in a program from the command line (programs will be read in as a sequence of 1s and 0s, but the function will ignore anything that is not a 1 or 0 allowing for comments).

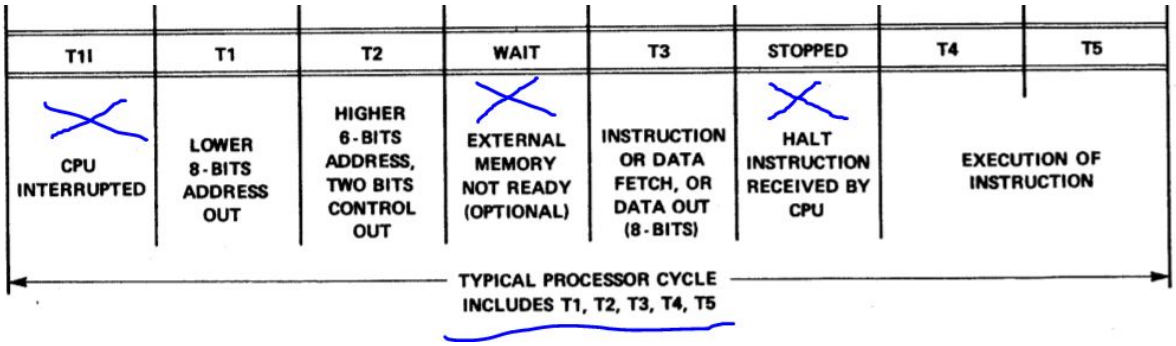
(b) Problems or issues encountered so far:

- Integration has been difficult.
- We need to determine how to do the pipeline without having to scrap our first implementation.
- Determining level of abstraction for each component has been difficult (e.g. do we need to create global variables or pass local variables, how closely does our code emulate the actions taken by hardware).
- The instruction decode, control, and main loop components have proved to be difficult to partition into separate workloads as the code is tightly knit between them.

(c) Proposed changes to the scope/deliverables of the project:

- We propose to implement the CALL and RETURN instructions if time permits, but we may not finish them. Note that our goal was to implement about half of the instructions, but we are implementing much more than that, so this isn't a change in scope, but it was worth mentioning.
- Because we may not implement the Call / Ret instructions we propose to only have one register to hold the value of the Program Counter instead of implementing the address stack.

- We have decided to implement only the typical processor cycle stages T1, T2, T3, T4, T5. The T1I, wait, and stopped states require simulating other piece of hardware. We believe these states are out of the scope of what we can



  accomplish in the time we have. This component of the project was not specified in our proposal, but with a better understanding of the processor, we thought it was worth mentioning explicitly.
- We will be implementing a HLT instruction, but rather than entering the STOPPED state, it will terminate the program (opcode = 0b11111111).