10-2: Met with group (15 min). We decided to either use a memory-to-memory architecture or a stack architecture
10-4: Met with group(1 hour). We decided on a stack architecture and began creating our ISA. We decided on three different types of instructions: A, B, and C. We then decided on which special registers to keep, along with a list of instructions that we need to implement.
10-6: Met with group(2.5 hours). Christian and Will worked on RelPrime code and convention call procedures while I worked on the instruction syntax and semantics so that all three of us would be on the same page. Some decisions I made about the instruction syntax is that

1. All instructions are lower case
2. Most instructions will be named similarly to MIPs instructions
3. The op-codes are organized by type

I also have begun identifying some pseudo-instructions/common operations that will make writing assembly shorter since it is extremely verbose right now
The biggest challenge was deciding on what variations of push to include so that we can load immediates on the stack and get/put values into memory.
10-7: Met with group(3 hours) . Christian and I discussed the ISA that we currently have and push/pop further. We then added ISA instructions and deleted some based on the discussions that we had with you.Will and Christian then discussed/finalized the calling procedures. I created/updated the op-codes for those instructions while Christian updated the RelPrime code to match our syntax. I went through and drew stack for Christian's code to check and see that it made sense. Will typed all completed code and I began translating the finalized code into opcode. We decided on the size of the register stack based on our discussion with you.
10-8: Met with group(30 minutes). We finalized the milestone 1 document and are going to push our stuff. We plan on looking through milestone 2 to make a plan on how to complete it over break. **End of milestone 1.**
10-9: Met with group (1 hour). Discussed feedback and decided what fixes to have done by the time we meet again. We got a rough outline of the memory allocation stack done and changed our reserved memory locations to reserved registers. We decided to add two new instructions, pushR and popR, to our ISA.
10-12: I worked on the project (45 minutes). I added pushR and pop R to our ISA, along with a pushM and popM. I then added some conditionals and loops to our common operations section, but ran into a problem with how to preserve our local variables, which was also brought up in our feedback.
10-13: Met with group(3 hours). We implemented our special sp register into our realprime code, which shortened the code by about a page and a half. We decided to keep the sp register at the top of memory stack and to store our local variables with a modified pushM and popM that adds onto sp with an immediate. Now the programmer can save local variables onto the memory stack and then deallocate the same memory once the function is complete. Christian completed what was left of common operations while Will fixed the realprime code. I began work on Milestone 2 by creating the execution stack component and listing all of the necessary components. I then started writing RTL instructions for all of our instructions. Eventually all three of us worked together to finish single-cycle RTL for all of our instructions except for flip and dupli. We then compiled questions for Sid that we still have for milestone 1 and 2. An important

side note is that we are still considering ways to implement our procedure conventions, and ran into a problem on how to save the return address in the exact order that we specified on the execution stack (return address on the bottom of the stack before the call) . We came up with either an instruction that directly changes the pc counter or using a label that we can push before the arguments. We will finalize that decision after discussing it with you tomorrow.
10-14: Went to office hours to ask questions the group decided on (15 min).Met with group(2 hours). I shared the answers to the questions I'd asked during office hours with Will and Christian. Will and Christian then finished the RTL code for flip, dup, sll, and slr while I started to work on what control signals we would need for our components. Will drew our basic component pictures while Christian and I filled out the rest of the information in component list. Christian and I finalized designing the execution stack component while Will compiled our RTL code into a simplified multi-cycle chart. We decided to finish our summary of how we checked our components tonight and to review/refine our document together tomorrow.
10-15: Met with Christian (1 hour). We fixed several small errors in our required hardware section and then created our process of checking the RTL code. We then turned in Milestone 2.
End of Milestone 2. We began to look at M3's documentation. We decided to begin the work by implementing our components. From our experience, it would probably be best to begin by designing the ALU, register unit, adder, and a mux.This makes sense because our memory, barrel shifter, execution stack, and register file are more complex. A good goal to have would be to have these units tested and implemented in Xilinx by Thursday.
10-16: Met with group (10 min). We divvied up the labs. I get lab 7 memory, Will has lab 6, and Christian has lab 8. We are meeting tomorrow 5-7.
10-18: Worked on memory lab 7 (1 hour).
10-21: Worked on memory lab 7 (3 hours). Completed the memory portion.Met with group(2 hours). Christian worked on making corrections on feedback while Will worked on ALU lab. Will and Christian divided their time creating components while I continued working on the memory lab and completed the control portion. We as a group plan on completing the register file and execution stack together. Worked on memory lab(2 hours). The memory and control components work separately but do not work together. It has to be a problem with how I am connecting them.
10-22: Worked on the project for a total of 5 hours for the day. Worked and finished memory lab 7(2 hours). Met with Will(3 hours). I fetched Christian's Xilinx components and went through/checked them in order to see how they worked and to see what in the design document needed changed. I found that the register file currently doesn't have our zero register set to where it cannot be written to. It currently is not always zero. Other than that, everything looked great. Will worked on fixing things from milestone 2. I created our memory component, but am unsure whether or not an address length of 16 will be too much for the FPGA board like lab 7 suggests. If so, we are going to have to adjust our pushM, popM, and program counter accordingly.
10-23: Met with group(2 hours). I demonstrated lab 7 and then completed the memory testbench testing for milestone 3. Christian worked on execution stack while Will began the datapath.We ran into an issue with whether or not popNum was a control or not, but we have decided that it is. Met with Christian(4 hours). I updated most of the components on the list,

created the control unit, described all of the control symbols and created the implementation plan. Will finished ALU testing and updated the component list. Christian finished testing on the execution stack. **End of Milestone 3.** We have decided to perfect our unit testing for the next milestone and to begin implementing our implementation plan after our meeting with you.
10-25: Worked on control list graphs(30 min).
10-27: Met with group (1.5 hours). We worked together on a control signals chart. We are confused about the timing of the execution stack since it is performing more than two operations at a time with our execution stack.
10-28: Met with group(1 hour). We discussed it with you and a major design decision has been made. Instead of popping off two values every time we do an operation, we are performing a peek. Once the operations on the peeked values are performed through the datapath, the value is popped back onto the stack and the other operations that occur inside the execution stack depending on the instruction. The execution stack deals with putting its final value on like so:

1. The stack always pushes one value onto the stack, what it pushes onto the stack is determined by pushSrc
2. The control signal popAmt determines how many pops to perform on the stack

Christian agreed to change our RTL and execution stack to reflect these changes. I have agreed to finish the multicycle control diagram. Due to tests and special events this week, we have asked for an extension on Milestone 4. I won't be able to work on the project until Wednesday due to seven hours of exams on Tuesday night and Wednesday morning.
10-30: Worked on state transition diagram (3 hours). We had to add additional logic and controls to our datapath in order for beq and bne to work properly. I also changed our RTL to always perform push/popM calculation so that our transition state diagram would not have an empty bubble. I then created as much as I could of our state transition diagram. I plan on finishing that and working on the control unit implementation tonight and having that done before our meeting with you as well as the first sublevel of the implementation plan. Worked on and finished transition state diagram (1 hour).
10-31: Worked on control unit component (3 hours). I am attempting to put the multicycle diagram into verilog, but am slightly confused on how to represent the four stages (fetch, decode, execute, and done). Talked to you and then looked at code on website to better understand.
11-1: Worked on control unit component (2.5 hours). I have begun testing just the control unit, but plan on adapting the .coe file so that I can test the control unit and memory together. **End of Milestone 4.**
11-3: Worked on control unit testing (1.5 hours). I am currently only using a testbench that sets one instruction and checking that the appropriate controls are set and states are reached. I am currently having issues reaching the correct state after the decode stage. For some reason, it is always the state that is 6 states above the desired one.
11-4: Worked on control unit (5 hours). The error from the day before was resolved after I asked you. I had forgotten to extend the number of bits on the current state and next state variables so it was cutting off my values as I was setting them. Once this issue was fixed, all of the instructions worked individually and set the correct control bits. It is worth noting that to complete the max number of states per instruction (5 states), the clock needs flipped nine

times. I then connected the memory and control modules, and then inputted machine code into the .coe file. I then tested a sequence of seven instructions.
11-5: Worked on the project for a total of 6.5 hours for the day. Worked on and completed sublevel test 1, e (4.5 hours). This subtest integrates the ALU, ALUout, Memory, and Memout. This subtest is essentially checking whether the popM and pushM instructions work properly without the execution stack or controls involved. I had trouble with making the timing on my tests for awhile, but I used an always block to make it easier. Worked on sublevel test 1, c (2 hours). This subtest tests that the shifter, sign extender, and zero extender can properly extend and shift the proper bits of the opcode from memory. I noticed a flaw and showed it to Will, who found a flaw in our original datapath that is fairly easy to fix. Essentially, we found that we need to add an additional sign extender component to the datapath so that we can input into the shifter, which takes sixteen bits in. I applied the fix in the partial implementation test.
11-6: Worked on project for a total of 4 hours today. Worked on and finished sublevel test 1, c (2 hours) from yesterday. After the change was made, the tests I conducted went smoothly. Began implementing the entire datapath (1.5 hours) with the control signals as inputs. Passed it off to Will and began going over the design document to update it. I discussed how we should implement IO and the sp register with Christian for about thirty minutes. We are compiling a list of questions for office hours today and tomorrow. **End of Milestone 5.**
11-7: Worked on project for a total of 9 hours today. Redid subtests in order to make implementation of datapath easier. Completely connected datapath and began datapath testing. A slight correction was made to our datapath was made in that one of the sources to mux ALUSrcB has to be sign extended.
11-8: Worked on project for a total of three hours. I changed our pcadder to increment by 1 because I did not realize that we are not using block addressing until I began hand-setting control signals for the datapath.
11-9: Worked on project for a total of eight hours. Will and I debugged some name and size errors on the datapath and then created a sequence of instructions to put through the datapath. Pushli worked perfectly the first time, but we had to correct our es_subsystem in order to get add to work. Specifically, I accidentally used the wrong alu instance which Will fixed. We then slightly changed our control sequence because the pushSrc needs set a cycle before the execution stack pushes it. I am now trying to completely get our branching instructions to work.
11-10: Worked on project for a total of five hours today. We are having random pushes and pops with our instructions, and I figured out it was timing issues between our registers and components. I met with the group for two hours and we changed it to where components are on the positive clock edge and registers are on the negative clock edge. This made our arithmetic instructions work. We were having problems with beq but we figured out that it needed split into more stages.
11-11: Worked on project for a total of 12 hours. We almost have all instructions through the datapath, and I have corrected our state diagram to reflect any changes made so that they can be applied to the control.v file for total integration. Will helped test some different sequences of instructions to ensure branching really works.
11-12: Worked on project for a total of 10 hours. I completed branch/jump instruction testing for the manual control datapath. I then updated the control unit to reflect the many changes that we

made to the multicycle transition diagram. I combined the control and datapath and struggled with the sizes of the controls for awhile. Once I began testing the testbench, I ran into several errors. All of these can be traced back to unnecessary resets I'd been making at the beginning of each current stage. Once I found this, everything worked. The combined datapath and control are officially confirmed for 10 instructions, but I am not counting arithmetic instructions. Will has been working on testing the others by manually setting the controls in the datapath, so hopefully we can integrate these tomorrow and get input/output completed.
11-13: Worked on project for a total of 7 hours. Finished adding and testing every single instruction through integrated control and datapath. Worked with Will to initialize the sp register properly and to update the assembler so that we could begin writing test programs faster.
11-14: Worked on project for a total of 9 hours. Began writing test code for our processor and also to familiarize with our assembler, including a multiplication function in order to see the best way to begin rewriting our relprime code. I then handed off the multiplication function to Christian so that he could try to get IO to work with a function. Will and I decided to write relPrime separately so that we wouldn't get stuck on the same things. Will completed relPrime before me and then began testing our processor specs while I tried to help Christian with IO as much as I could. Will jumped in as well once he was done with timing. We had multiple problems with version control of the project tonight, which slowed us down a bit.
11-15: Worked on project for a total of 1.5 hours. Helped Christian as much I could so that IO would be complete after getting help from you. **End of Milestone 6.**
11-17: Worked on project for a total of 8 hours. Helped create comments and waveforms for repo. Worked a lil on design document. **End of project.**