Team 3v: Design Process Journal

Members: Maura Coriale, Ben Gothard, Matthew Lyons, Joy Stockwell

Milestone 1:

Meeting 1: 1.5 hours
September 28, 2018
Members Present: Matthew, Joy, Maura
Overview: We began the process of creating our instruction set architecture by deciding on creating a mostly accumulator-based CPU with some facets of stack. We then began brainstorming our commands and instruction types, as well as deciding conventions for procedure calls, instruction layouts, etc.
We also began writing our first draft of the Euclid's method program using our instructions and current syntax so far, and Joy offered to write a complete second draft before our next meeting.
Major Decisions: Accumulator type (with elements of stack), instruction types, commands, and conventions.

Meeting 2: 1.5 hours
September 29, 2018
Members Present: Matthew, Joy, Maura
Overview: We picked up where we left off last night, starting with going over Joy's second draft of the Euclid's method in our own assembly. We then changed some of our calling conventions that we decided on yesterday, and discussed how comparisons and jump commands would work with our processor. We also worked to try and fix some problems we came across with our commands.
Major Decisions: "Frankenstein type" of accumulator with Stack, calling conventions with procedures,

Meeting 3: 1.75 hours
September 30, 2018
Members Present: Maura, Joy, Matthew, and Ben
Overview: We met again after we all worked on our own portion of Milestone 1. While working on code fragments, Matthew found a

problem with how we are using stack. Ben proposed a new way to approach the problem, including a possibility of 2 accumulator registers to handle problems with operations during function calls. Maura proposed a flag bit for modifying existing instructions. Currently, we are tackling the question on how much the compiler and assembler should do for us and how much should be implicitly in the language. Ex: OP Code for add and lw are the same, only func code is different, should we do something similar with our flag bit?
The Accumulator Problem: Since we probably want 2 accumulators minimum, how do we use them, explicitly by calling them by name in the code or implicitly building it into the hardware with multiplexers and having a flag of sorts that comes from the accumulator that tells the CPU whether a particular accumulator is empty and then makes the decision about where to put the value from there. We whiteboarded side-by-side several different problems (a simple addition problem and a summation in a for loop) to try and see some pros and cons of each solution.
We continued adding to our list of questions to ask Sid on Monday during our meeting.
Major decisions: Needing multiple accumulator registers, need to change how we interact with data, Implicitly or explicitly moving values between accumulators

Meeting 4: 2.25 hours
October 1, 2018
Members Present: Maura, Ben, Matthew, Joy
Overview: Chose a two-accumulator with a backup accumulator as a compromise between Ben's and Matthew's ideas for the accumulator design. Flag bit for choosing whether to do functions between the accumulators or the stack. All values are signed. Continued making Frankenstein-based name puns (Instruction set = Clerval, Main accumulator's nickname is Mary, secondary's nickname is Shelley). Decided to reduce to one instruction set type to simplify the assembly process, and Ben started writing a basic assembler in Python. Worked through problems as we came across them in writing the assembly fragments, assembling them, and working out syntax and semantics.

Major Decisions: Calling conventions cont, main and backup accumulator, flag bit

Total for Milestone 1: 7 hours

Milestone 2:

Meeting 5: 1 hr
October 3, 2018
Members Present: Maura, Ben, Joy
Overview: Decided on meeting times for the next milestone, split up the RTL commands to write, and discussed some possibilities for I/O and opinions on single/multi-cycle/pipeline.
Major Decisions: NO PIPELINE

Meeting 6: 0.75 hr
October 5, 2018
Members Present: Maura, Joy, Matthew, Ben
Overview: Reviewed our individual RTL's, decided to meet again Saturday morning and evening to try and finish up Milestone 2.
Major Decisions: RTL conventions

Meeting 7: 1.5 hr
October 6, 2018: 1.5 hr
Members Present: Maura, Joy, Matthew, Ben
Overview: Met in the morning to start working on State Diagrams, figure out our shopping list of computer components, etc. before our meeting in the evening, due to time conflicts on other days.
Major Decisions: Some conventions for RTL, wrote out State Diagrams

Meeting 8: 1.5 hr
October 6, 2018 (pt. 2)
Members Present: Maura, Joy, Matthew, Ben
Overview: Joy and I worked more on State Diagrams while Matthew and Ben began drawing a circuit diagram to start on the shopping list of parts needed and their states and stuff.

Notes: Want to left shifter hooked up to B, in both the stack and jump to function, going to be left-shifting immediate and storing in b
Major Decisions: Decided to change our flagbit to be part of the mnemonic, so for aadd with accumulators, for example, it would be aadd@, while an aadd with

Meeting 9: 1 hr
October 7, 2018
Members Present: Maura, Ben, Joy, Matthew
Overview: Worked individually mostly, but hashed out what was left to get done on Milestone 2.
Major Decisions: Interrupt-based IO

Meeting 10: 1 hr
October 9, 2018
Members Present: Ben, Joy, Matthew
Overview: Created tests for the RTL, fixed assembler to support flagbit decoration and added label support
Major Decisions: Decided to test the RTL by tracing through it and noting the status of each register before and after each multi-cycle chunk. Also added a no-operation instruction (noop) which is used by our assembler to implement loops.

Meeting 11: 0.5 hr
October 9, 2018: (w/ Sid)
Members Present: Maura, Ben, Joy, Matthew
Overview: Met with Sid, discussed Milestone 2 and our approach for starting Milestone 3
Major Decisions:

Milestone 3:
Meeting 12:
October 15, 2018 (in class)
Members Present: Maura, Ben, Matthew, Joy
Overview: Split up work for the 3rd milestone, recapped what needs to be done in Verilog and what can be done by hand, along with priorities and in what order to do things in.
Major Decisions: N/A

Meeting 13: (in class)
October 16, 2018
Members Present: Maura, Joy, Ben, Matthew
Overview: split up the work and mostly worked individually
Major Decisions: Adder--It turned out we could just use the + in verilog for our adders. If we end up not wanting to do that, we can also use ready-made adders from Xilinx. Either way, we won't end up needing to do much testing.

Meeting 14: (in class)
October 17, 2018
Members Present: Maura, Joy, Ben, Matthew
Overview: Continued working on individual pieces, along with collaborating to solve problems and create the data path
Major Decisions: Specifics of the data path

Meeting 15: w/ Sid
October 19, 2018
Members Present: Maura, Ben, Joy, Matthew
Overview: Discussed the milestone with Sid, reviewed our status on Milestone 3 and saw where we were behind and what we needed to improve.
Major Decisions:

Milestone 4:
Meeting 16:
October 20, 2018
Members Present: Maura, Ben, Joy, Matthew
Overview: Worked on our areas for improvement identified by Sid in our last meeting. Rehashed the integration plan, focusing on a more layered approach, and describing how we created each of the components in detail. We also started figuring out Milestone 4.
Major Decisions:
N/A

Meeting 17:
October 22, 2018

Members Present: Maura, Ben, Joy, Matthew
Overview:
We discussed I/O and extra features
Major Decisions:
Output -- automatic
Input -- interupts, checking at the beginning of every instruction, writing to Mary
Exceptions (if we get to them) -- check at the beginning of the cycle that would write back to registers, and don't write back to registers if there is an exception; hardcode error messages into kernel
FPGA board -- Joy, with help from Ben and Maura, will put on FPGA board if enough time on last weekend
Assembler -- already basically written by Ben
We didn't know if we would get to exceptions and putting on the FPGA board. We decided to make them lower priority because nothing else depended on them. We decided we couldn't implement interrupts without help from all the group members (was originally going to be only Joy's responsibility) because it was a large commitment with far-reaching consequences. We decided as a group to go through with them. We decided that, to make up for the added difficulty, we would go with the simplest form of output: automatic.

<u>Meeting 18:</u>
October 22, 2018
Members Present: Maura, Ben, Joy, Matthew
Overview:
We established hardware to support interrupts. We started integration.
Major Decisions:
We would need a coprocessor and an extra bit going into control. We would keep the kernel at the beginning of memory. We changed our integration plan to integrating in slices instead of onion layers. This way, we could all start integrating part without waiting on anybody else's work.

<u>Meeting 19:</u>
October 23, 2018

Members Present: Maura, Ben, Joy, Matthew
Overview:
We worked on integration and control. We hit a bit of a wall getting pc set to pc + 2. We delegated work for tying up loose ends for the milestone tomorrow.
Major Decisions:
Matthew and Joy were to put integration on hold and write pseudo-code for tests of not-yet-integrated parts of the CPU.

Meeting 20:
October 24, 2018
Members present: Maura, Ben, Joy
Overview:
We tied up loose ends. Matthew was in another class so he could not attend.

Meeting 21:
October 31, 2018 - Spooky Scary Xilinx Problems
Members present: Maura, Ben, Matthew, and Joy (all half-alive)
Overview:
After working a lot over the past few days to finish up our individual integrated components, we finally got all tests working and started work on creating our masterpiece, Frankie (coincidentally this is occuring on Halloween). With most of the integrating done, we troubleshooted issues with our integrated PC/SP block.