Brendan King Work Log

CSSE232-03

Final Project

Contents

Milestone #1:

1/12/2020

3.5 hours:

All four group members met. We decided to make only two instruction types to keep things simple. The first bit in each instruction determines the instruction type. All instructions are only 16bits long to increase simplicity and to reduce memory needed to store programs. We created a base instruction set. We created a few novel instructions including branch if negative (bin). Bin was created since we decided set less than in the MIPS architecture was mostly used in the context of branching, so we thought bin would reduce the number of instructions.

1/13/2020

30 minutes:

All four group members met. Jump and memory access commands now use the address stored in R1 instead of always using register 15 so that extra instructions don't have to be used to always move the address to $15. GCD code written by Jeff was presented. Tasks assigned to do before next meeting:

- Jeff – type up GCD code
- BK – add code to handle stack to GCD code
- Chet – Begin translating assembly to machine code
- Aurora – Add missing elements to the Design Doc and format it

40 minutes:

I added code to handle the stack to Jeff's GCD code. I also wrote a code snippet depicting how a basic for loop works. I also took the time to ask one of the T.A's about planning for I/O features.

1/14/2020

45 minutes:

I carefully read over Jeff's GCD code on the design document and added comments. I found what I think is one error in the code. I also typed and commented my for loop code I created yesterday. I added example code that contains conditionals and how to load addresses.

30 minutes:

Everyone met for 30 minutes to confirm the GCD code and assigned tasks for the next meeting.

Things for people to do before next meeting:

- BK – convert programs to machine code (possibly with assembler)
- Chet – make example machine code for each instruction
- Aurora and Jeff – Format the design document and ensure that every requirement has been fulfilled.

45 minutes:

Took the time to finish Chet's assembler program so that it can support multiple instructions and comments. It still needs to be debugged.

1/15/2020

1 hour:

Finished debugging the Java assembler. It is not as robust as I would like it to be but it is good enough to assemble all the code on the design document without error. The code has been uploaded to our Microsoft teams folder.

30 minutes:

The entire team met and finished formatting the Design Document. To begin M2 we assigned the following tasks that should each take 15-30 minutes to do by tomorrow:

- Jeff and I – Spend time getting comfortable with Xilinx and possibly build something simple like a mux
- Chet and Aurora – Begin writing RTL fragments

Milestone #2:

1/16/2020

25 minutes:

I read the Verilog one-day tutorial in the resources section of the course webpage and took notes.

1/19/2020

3.5 hours:

We met and created a new instruction set that does not directly address any registers. The only register besides the accumulator is a stack pointer register, but the programmer is only able to increment and decrement that register. We wrote some example assembly code. We decided to add a right arithmetic shift to make division of negative numbers easier. I brought up a point about how difficult it may be to implement a shift by an arbitrary amount versus a single shift operation. We will ask the professor for advice.

1/20/2020

25 minutes:

I revised most of the assembler code to make it work with the new instructions. I also came up with the idea to revise the spa instruction so that it moves sp by the value at sp+imm so that a value can be carried by ACC to other parts of memory. I also want to ask the group if they want the assembler to support variable names so that the programmer does not need to remember memory address locations.

30 minutes:

Everyone except for Jeff met and worked on finishing what the design document needed for M1.

1/21/2020

40 minutes:

I finished the Assember for the new instruction set and converted the gcd assembly code to machine code. Formatting it on the word document took a while.

30 minutes:

We all met except for Aurora in the library. We wrote down the RTL we need but we still have yet to double check it. Before next meeting, we are to each come up with hardware component inputs and outputs lists.

1/22/2020:

2.5 hours:

I reviewed the GCD code and discovered that it incorrectly addressed locations on the stack (most of the negatives were flipped among other problems). I went through and fixed the errors and re-assembled the code. I also put in the recursive code I wrote at one of the meetings and assembled that. I also took the time to write out inputs and outputs for most of the components that we need.

1 hour:

We met as a group and finished formatting the design document and rereading and revising the RTL. We decided to combine 2 cycles in the lw operation RTL since one of the two instructions just moved one value from one register to another.

Tasks for next Milestone:

Begin implementing hardware we outlined on the document (at least a register or mux).

Milestone #3

1/23/2020

10 minutes:

Wrote down notes of everything that the professor told us to improve on our document.

1/26/2020

1 hour:

We designed the datapath for the processor and traced a few commands through the datapath to verify that it will work. We decided to get rid of the zero extender, since the ALU could take care of only looking at the last 8 bits of the data if necessary.

1/27/2020

15 minutes:

We met to discuss progress on the labs. We will try to finish as much as possible tonight so that we have more to collaborate on tomorrow.

1/28/2020

1 hour 30 min:

I finished and tested the memory and control for lab 7. I have yet to combine the two to finish the lab.

10 min:

We met to discuss our progress.

30 min:

I worked on writing implementation and testing plans for different components.

1/29/2020

30 min:

I finished writing implementation and testing plans for different components.

3.5 hours:

We worked together to finish out the milestone. I also made an implementation of distributed memory for our project. We decided to use distributed memory since it is faster. We decided to make the extender zero extend or sign extend based on a control bit to reduce the amount of logic needed to be implemented by the ALU. We also decided to create a new piece of hardware called the BLU (Branching Logic Unit) since there were many types of branches and it helped to compartmentalize that logic in our design.

I did not find any errors in the implementation of memory that we used. When I ran into problems it was usually because of some mistake in the testbench itself and not the actual hardware.

The most major effect that the accumulator architecture had on our design was that it forced us to not use a register file. Although not very significant, due to our accumulator architecture, our mux going into the ALU had to be larger in order to select from the sp register in addition to the ACC and PC registers. Most other changes were also not very significant.

When we created tests we wanted to exhaustively test every single type of input and edge case without writing so many tests to the point that all our time is spent on testing instead of building. Thus, when we test, we want to test cases with standard input (like positive and negative numbers in the case of the ALU) and edge cases that we might not have covered (like inputs of all zeros or all ones).

Milestone #4

2/1/20

1.5 hour:

We all met and we planned out what needed to be done for this milestone and build the state machine diagram.

2/2/20

1 hour:

We met in class and I worked on building the memory and control for the project. I changed some of the opcodes for commands to reduce the logic in the control. Now for certain opcodes and states, the next state is the opcode itself.

2/4/20

4.5 hours:

I had to write case statements for the control for every single command based on the state diagram and then make tests for much of the cases. I also began the first step in the integration plan and wrote tests for that integration plan.

2/5/20

1.5 hours:

We all met and finished editing the design document and committing work. I am very close to finishing part 1 of integration.

Milestone #5

2/9/2020

2 hours:

I worked alone to finish testing and debugging the implementation step one. There was nothing wrong with the implementation, I just took forever to get the tests to work.

1 hours:

We met to plan and discuss what our goals over the next week are. We looked over the two parts of the implementation plan that have been completed.

2/10/2020

1 hours:

We met in class. Chet and I drew a schematic connecting the two halves of the processor that have already been made.

1.5 hours:

Chet and I met outside of class to finish the schematic and to begin figuring out how to debug the finished processor.

2/11/2020

1 hour:

I continued to write short assembly snippets to be converted to machine code to test each instruction the processor supports.

2/12/2020

1 hour:

I finished assembling code for the immediate instructions and finished debugging them using the simulator.

3.75 hours:

We all met in class and worked together until 4:45. Chet and I worked together to debug lw, sw, in, out, spi, spc, lwa, and the remaining arithmetic instructions. We helped to format the design document before turning it in for the week. We still need to debug the jump and branching instructions before all instructions have been debugged. We found out that some of the control bits that we thought we don't cares for lw and sw actually needed values for those cycles.

I ran into problems using the assembler for the li pseudoinstruction when the immediate is negative. I will need to fix the assembler.

We are almost done with the processor. After finishing debugging individual instructions, we can try running our GCD program through the simulator. After this, we can try connecting the peripherals that

Geoff and Aurora have been working on and try uploading our work to the FPGA board. If after this we still have time, we should make a compiler.

Milestone #6

2/16/2020

2 hours:

Chet and I worked on finishing individual tests. We finished the jumps and branch tests. We found that we needed to add a temporary register to hold the value of the PC during jal instructions for an extra half clock cycle.

2/17/2020

1 hour:

Chet and I worked on testing the sum program written in the design document. During this time, the test did not work. Aurora and Geoff have been working on I/O.

1 hour:

Chet and I continued to debug the sum test. We managed to get it to work after changing one instruction.

2/18/2020

1 hour:

Chet and I worked on uploading the GCD program to the board and testing it.

2/19/2020

4 hours:

Chet and I worked on finishing debugging the GCD and relprime programs. A few lines of code were changed because there was confusion about if spi shifts its immediate. The description of spi in the design document was changed to make it clear that spi shifts its immediate.