4/3/2022

Go through some code, decide procedure calling conventions.

TODO:

Text

Description automatically generated

4/6/2022

Adjust the instructions to fit into 16 bits.

Set all instructions to be 0x00XX and data to be 0xFFXX

Replaced $ra with a specific address 0xFFFE

$sp set at 0x1FFF

4/7/2022

Meeting with the professor

Write minimum code for the procedure call

Add the caller of relPrime

Write an assembler

4/8/2022

Finished writing the RTL

Thinking about changing jump or changing sw lw to make them into 4 cycles

4/16/2022

Tried writing the assembler using java

Get file not found error

Plan to try using Python

4/19/2022

Plan not to change the addressing mode, since sign extent is powerful in a way that it does not need the ALU, may add an addressing mode that uses ALU when we switch to pipelining

Go through the document of M2 and tell Zeen what to do.

Thinking about having a weekly team meeting time

4/20/2022

Zeen is available Sunday so thinking about 2 pm Sunday.

Current Roles:

Graphical user interface, text, application, email

Description automatically generated

For Zeen:

Text

Description automatically generated

Zeen takes over Assembler

Thinking about adding pseudo instructions and improving RTL. Think better leave it like that since there is a possibility that we will switch to pipelining. Not may possible pseudo instruction, since we will need to have an extra register or store the value of acc on stack then retrieve it, maybe better without it.

Pseudo Instructions:

sw 0(m): load m, sw 0

Store m on stack with offset 0

slt a,b : load a, slt b

Compare a and b, if a<b, put 1 into acc, else put 0

addi m 1: load m, addi 1, save m

Increment m by imm

We do not have a way to address registers other than acc for most instructions. Can only do the 2nd method.

4/22/2022

Half finished with building datapath.

Added a cycle for sw and lw and do the pre-calculation for beq, bne instead since they are used more often

Tried to use the unused bit to switch between the two but beq, bne are AI type, they do not have the 3 extra bits.

If we are changing the source for ALU, we to choose between the op of the mux to be from IR or control. That means adding a mux to the opcode and using another bit from IR to be the op of this new op mux. But we cannot switch back.

TODO: Add the memory map, increment jump to use 11 bits

TODO: Finish datapath on Sunday

4/24/2022

Finished designing the datapath with Jermaine

Made minor changes to the RTL

Text

Description automatically generated

Errr.. just noticed that my role should be the test implementation and description, maybe that causes the confusion?

Memory map, update j, jal green sheet and RTL

4/26/2022

Finished updating changes to j jal. Finished adding memory map.

Write some plan of testing the implementation, but the implementation plan itself is not yet written so may need to make changes to it.

The test I write basically focused on testing all the possible inputs of the control signals. Tests for flags and clock are also added for those that have these.

Decide to set memory to read and write on rising edge since in jal we want it to store the current PC before PC changes on falling edge. Does not have a cycle where memory do both read and write so it is okay to put read and write on the same edge.

Main should start at 0x0000

ALU op should be 3 since it does add, sub, and, or, slt

Finished the Complex FSM for M4

TODO: For M4, finish implementation of test bench, need to contact Zeen to see if he can meet during Thursday Friday class time.

TODO: Ask Zeen to use the assembler to get the new machine code that has address of main starting at 0x0000