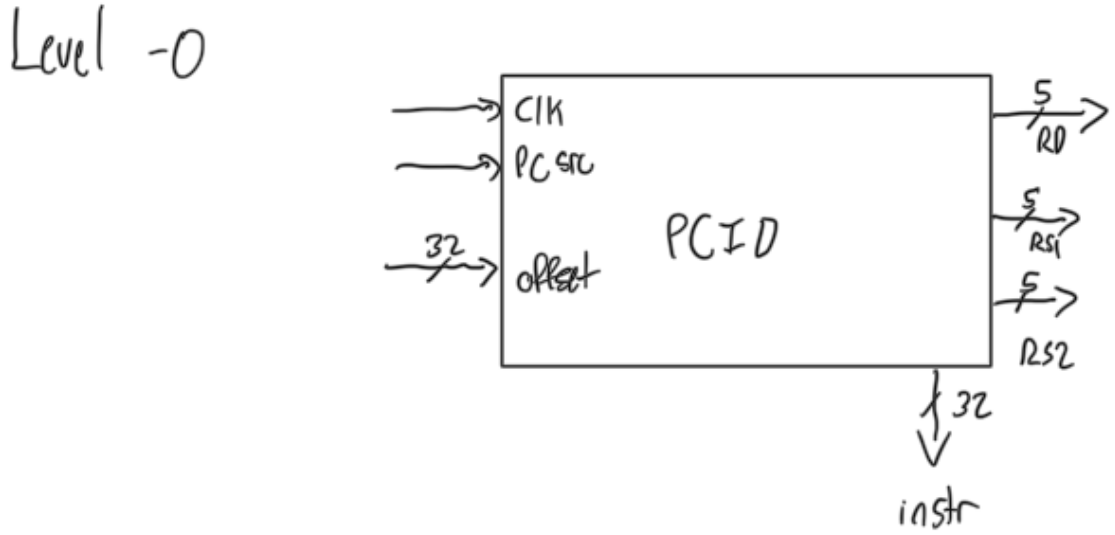
Lab 7 — Program Counter and Instruction Decoder

Objectives
For this lab, I am creating a program counter and instruction decoder for a processor which uses RISC-V ISA. It will also be connected together with instruction memory.

Introduction
Two essential parts of a computer processor are the program counter and instruction decoder. The program counter in a processor is what keeps track of the address of the instruction being executed in a program at any time. The instruction decoder reads the instruction at the address in the program counter and decodes it into components to be sent to their different places.

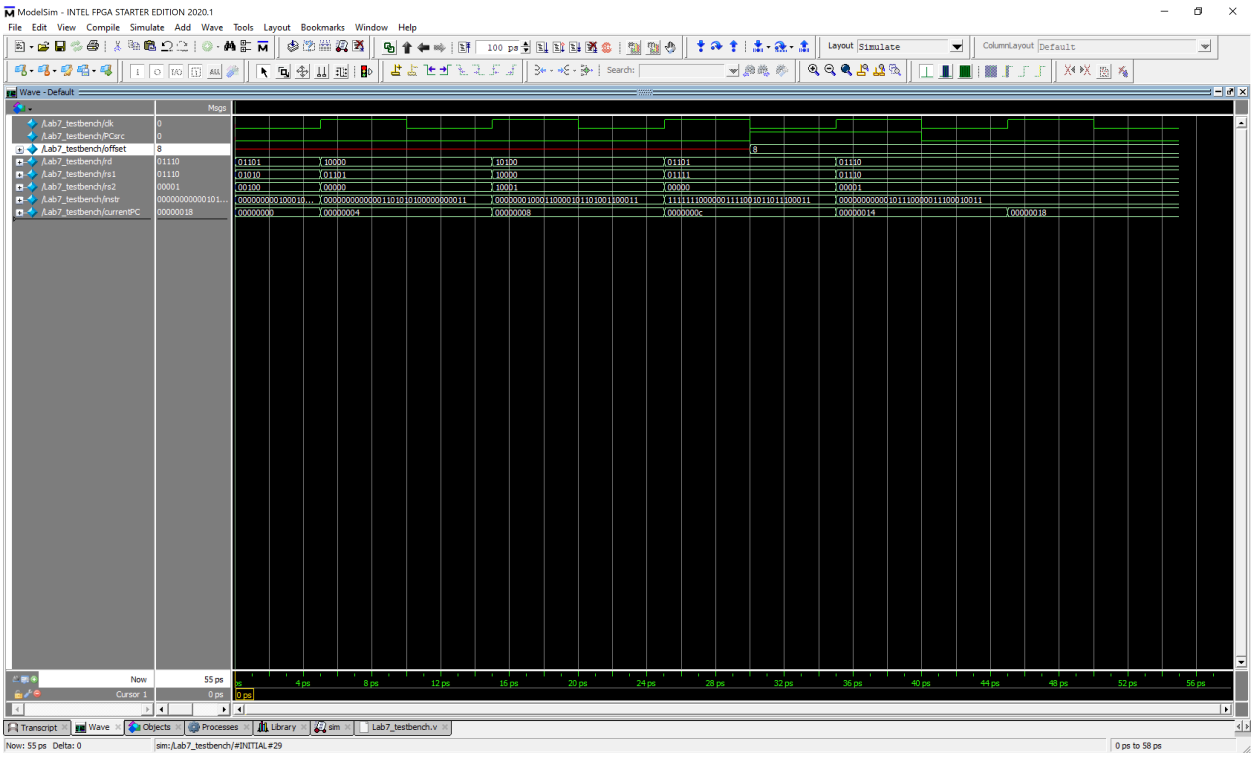
Methodology
The following is a level -0 diagram of my program counter and instruction decoder. It has a 1 bit source input, clock input, and 32 bit offset input. Based on the source input bit, on each positive clock edge the program counter will increment either by 4 or the input offset value. For any current program count, the instruction decoder portion outputs 5 bit RD, RS1, RS2, and the 32 bit full instruction from the built in ROM.



The program counter has a register for the current program count, as well as one for the next program count. On the positive clock edge, the current program count is set to the value of the next program count. The next program count is determined by either an adder adding 4 to the current program count or one adding the offset input. Which is set as the next program count is determined by a 2:1 mux whose input is PCsrc.

Result and analysis

When tested in ModelSim, the program counter and instruction decoder properly increment through each instruction in the ROM. The offset input and PCsrc inputs also work properly to increment by a value other than 4 when required.



All code, diagrams, and simulation results can be found as files on GitHub: https://github.com/Eth7an/Lab-7.git

A video explanation of the top level design and ModelSim simulation can be found here: https://youtu.be/nNdOteNMK_w

Discussion and Conclusion

To conclude, I have built a fully working and tested program counter and instruction decoder module. The functionality of all inputs and outputs has been tested and verified, and this module is complete and ready to be instantiated in higher level projects, such as a full CPU datapath.