EC413 - Lab 6
Siara Patel & Vanshika Chaddha

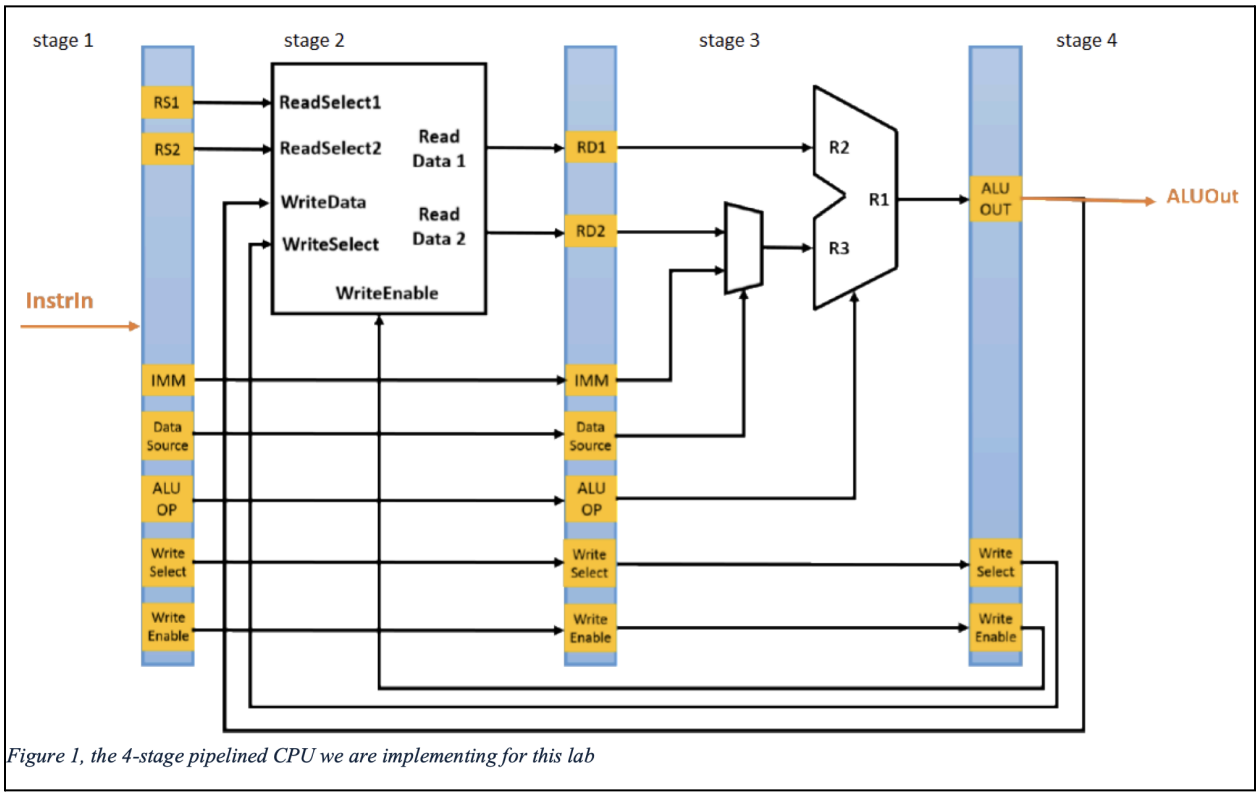


Figure 1, the 4-stage pipelined CPU we are implementing for this lab

Source Files:

- Register_File
- R1_Register
- R2_Register
- R3_Register
- ALU
- MUX
- Pipeline

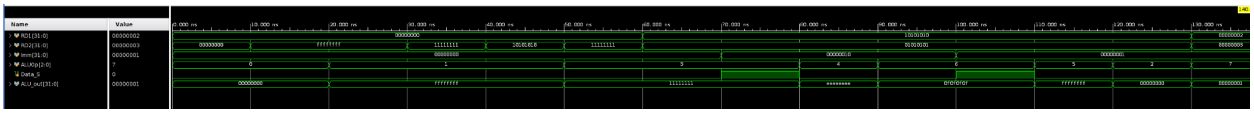
The previous sources implement the 4-stage pipelined CPU illustrated above. The register file implements the logic of the block in stage 2 that receives the read selects, write data, write select, write enable, and outputs the contents of read selects. For example, if the read selects are 1 and 2, this file looks into registers 1 and 2 and outputs its contents.

R1_Register, R2_Register, and R3_Register each pass the data along the pipeline and are dependent on the clk. In the image above they represent each of the blue strips. For the R1_Register, the input is the encoded InstrIn. This module then decodes the InstrIn and assigns RD1, RD2, IMM, Data Source, ALU Op, Write Select, and Write Enable. R2_Register works similarly to R1_Register and passes all the needed inputs to the next stage of the pipeline.

Finally, R3_Register takes in the output of the ALU module, write select, and write enable. It only outputs ALUOut, the result of the CPU pipeline.

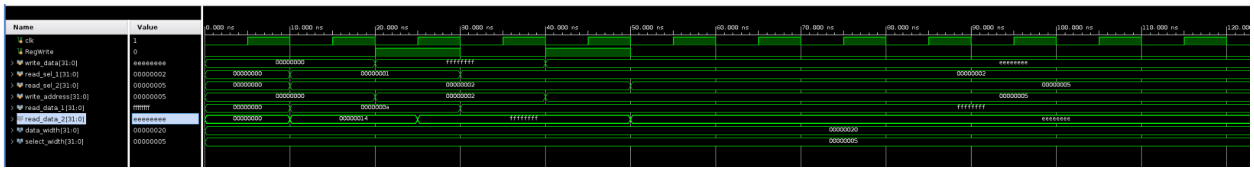
In the pipeline source, all of the other sources are instantiated to build out the entire pipeline. Starting with R1_Register, wires are used to capture the outputs of this module and passed to the next stages, the Register file module and R2_Register. Then another set of wires are used to capture the output of the R2_Register and passed to the MUX, ALU, and R3_Register. The Data Source signal, either a value of 0 or 1, is set based on the instruction set, R-Type or I-Type. If the Data Source is 0, the instruction is R-Type, and the ALU utilizes RD2 to compute. If the Data Source is 1, the instruction is I-Type, and the ALU utilizes the Immediate signal to compute. Finally, the output of the MUX is passed to the R3_Register.

Generated Waveforms:
ALU Testbench:



How we tested: Implemented different cases for opcode type (different ALU operations) and values of A and B to test and checked the output to ensure it matched what was the expected ALU computation result.

Register File Testbench:



Checked to make sure output matched expected values.

Pipeline Testbench: