CECS 530 - Lab 3

"Register File"

Due date: 09/29/20

Student Name: Neha Bhoi

Student ID: 026627560

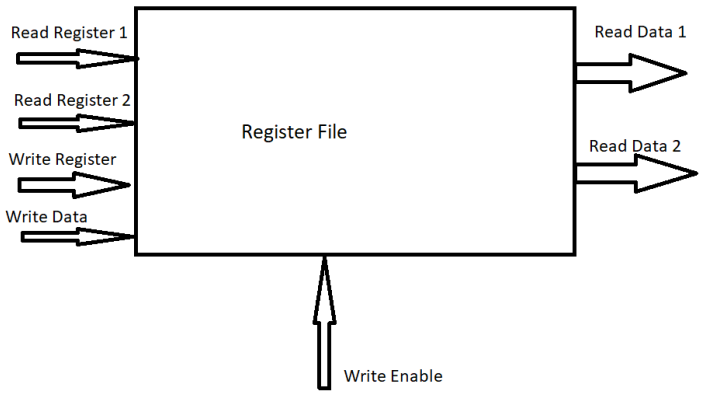
I certify that this submission is my original work

N. N. Bhoi -

Your signature or electronic signature

Lab Report: Lab Assignment 3 - "Register file"

1. **Goal:** The goal of this lab assignment is to simulate Register file which has collection of 32 registers. The size of each register is 64 bits wide. Also, we have to support R type ALU Instruction. A register file is an array of processor registers in a central processing unit (CPU). Refer below figure, depending upon the value of control signal (write enable) we can read from register file or right to register file.



2. **Steps:** Below are the steps involved in this lab assignment:
   - **a.** Design register file using register address inputs rd_addr_1 , rd_addr_2 and wr_addr 5 which are 5 bit wide, data input 64 bits wide and input control input. It will store the values in 32 register.
   - **b.** Design combine RF with ALU. Where ALU read data from register file and perform operation on it.
   - **c.** Design write back logic which will store the ALU result back to register file.

3. **Results:** The result of the lab is shown in the below screen shot of the simulation.
    a. Simulation of register file

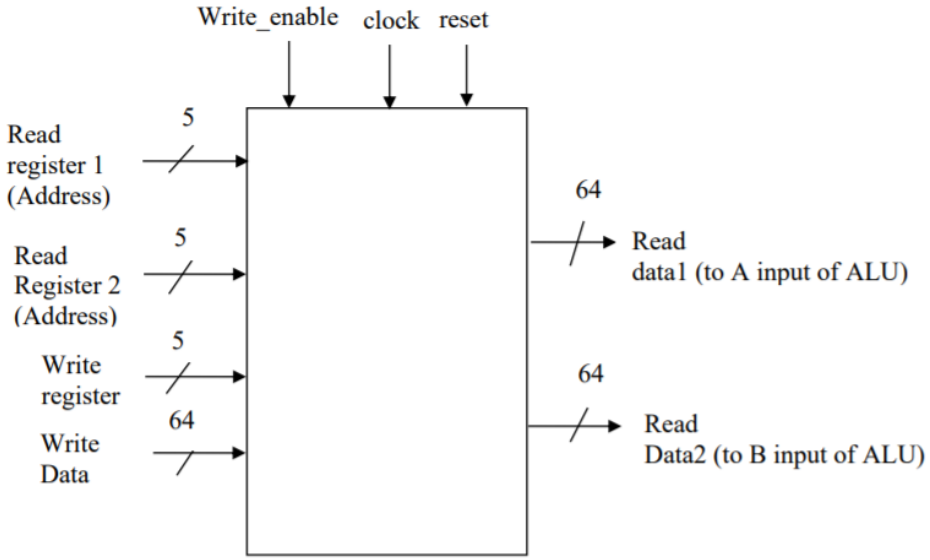
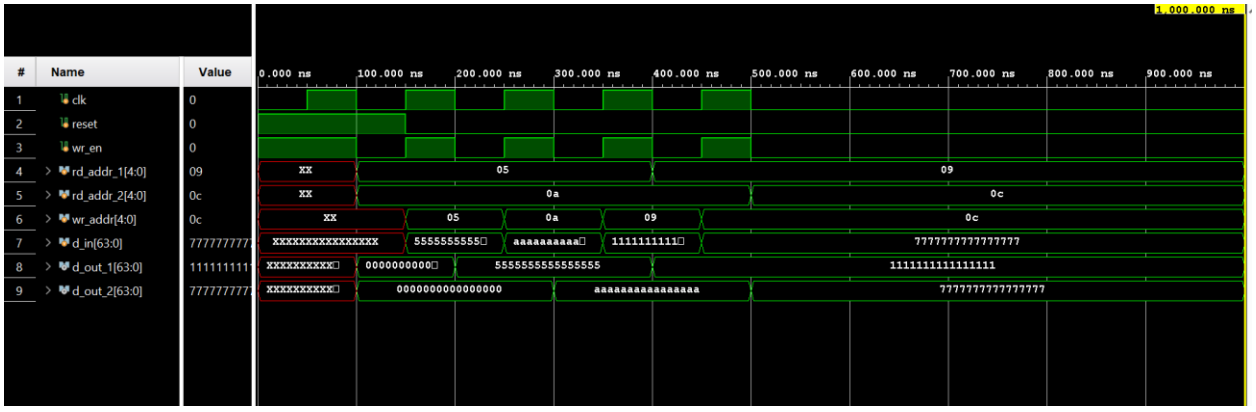


Figure 1. Register File

Register file operation parameters are:

- Input clk, reset, wr_en 1 bit
- Input rd_addr_1, rd_addr_2, wr_addr 5 bit
- Input d_in 64 bit
- Output d_out_1, d_out_2 64 bit

Below screen shot show the waveform for input and output parameters of Register file.



Here we are creating register file of 53 registers hence the address inputs are 5 bits. First, we are resting the register file by enabling the reset signal. We are setting rd_addr_1 to 5 and rd_addr_2 to 10. Then we are writing 64'h5555555555555555 to register 5 on positive edge and 64'haaaaaaaaaaaaaaaa to register 10 on next positive edge. In the testbench we are first setting the value of register address with data input and the changing the clk value so that correct data will be store in the register. And correct data will be read on next clock edge.

b. Simulation of register file with ALU

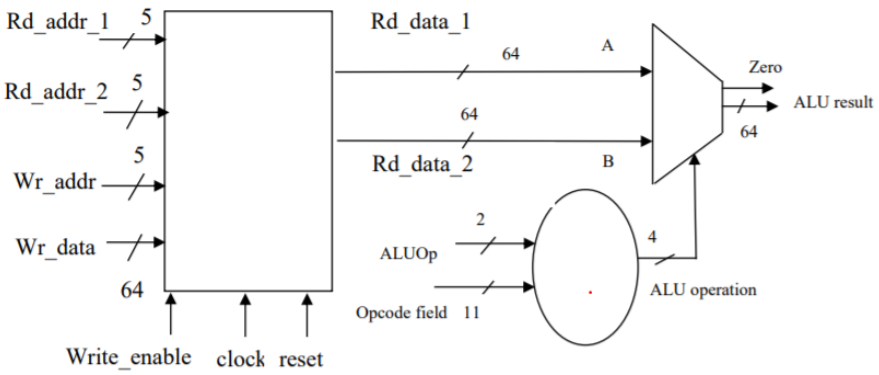


Figure 2. Register File feeding ALU

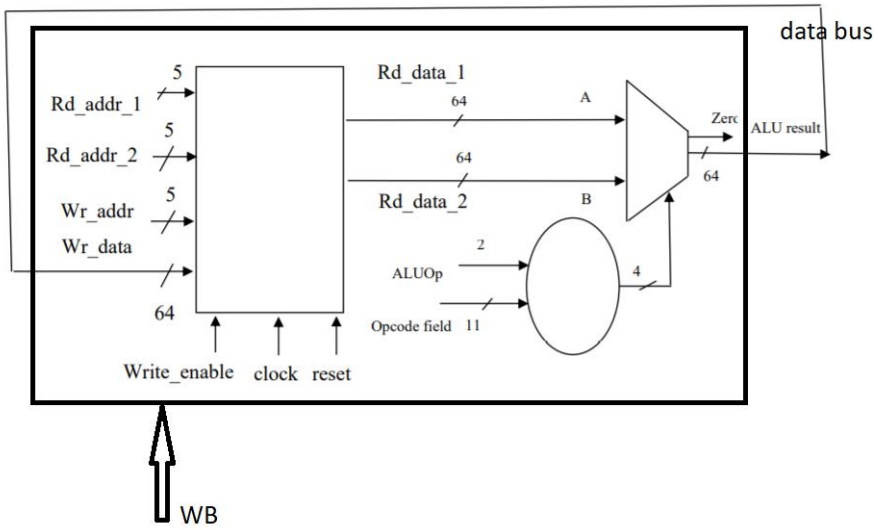
Register file with ALU operation parameters are:

- input ALUOp 2 bits
- input Opcode_field 11 bits
- input clk 1 bit
- input reset 1 bit
- input wr_en 1 bit
- input rd_addr_1, rd_addr_2, wr_addr 5 bits
- input d_in 64 bits
- output ALU_result 64 bits
- output Zero 1 bit

Below screen shot show the waveform for input and output parameters of Register file with ALU.

Here we consider register 5,10 as A, B and assign value 64'h5555555555555555, 64'haaaaaaaaaaaaaaaa respectively. The output of register file is the input to ALU unit. Based on opcode and ALU opcode value we perform ALU operation on the data stored in register 5 and 10.

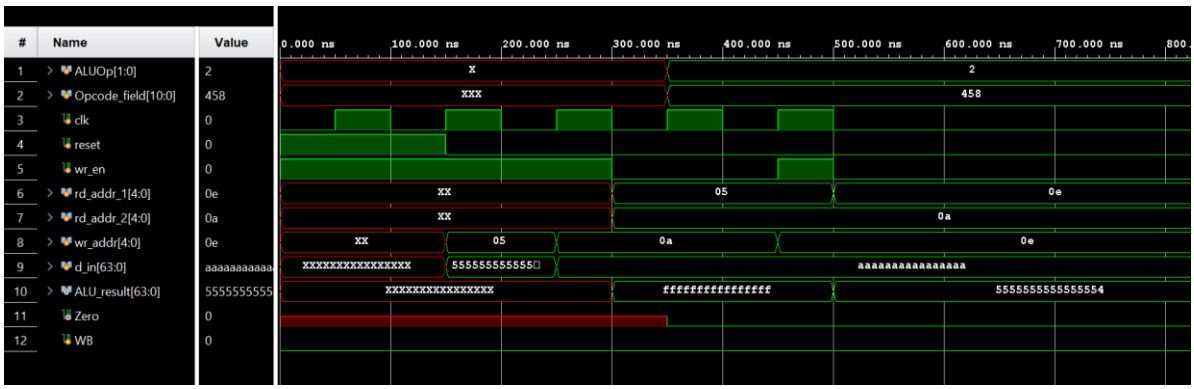
c. Simulation of Write back.



Register file with ALU and write back operation parameters are:

- input ALUOp 2 bits
- input Opcode_field 11 bits
- input clk 1 bit
- input reset 1 bit
- input wr_en 1 bit
- input rd_addr_1, rd_addr_2, wr_addr 5 bits
- input d_in 64 bits
- output ALU_result 64 bits
- output Zero 1 bit
- input WB 1 bit

Below screen shot show the waveform for input and output parameters of Register file with ALU and write back.



Here we have introduced one input called write back. If the this signal is set the module is loading the value at ALU_result output to the write register specified by wr_addr. This implementation is simulation R-type instruction. In the teste bench we are simulation below instruction :

ADD X1, X2, X3

Where, 5 and 10 are source register and 12 is the destination register.

4. **Conclusion:** This assignment was an introduction to the basic implementation of register file with ALU and write back feature to support R-type instruction. I have learned how to create array of register in Verilog.