1 LABORATION 2 – SEQUENCIAL CIRCUITS

In the laboration, we build a register file that will be used to store data in the microprocessor. We will describe the design using RTL-modelling in System Verilog. We also exercise how to build a basic testbench to check its functionality. We will also update the ALU that we designed in the previous lab to register its outputs.

The lab only requires simulation in Questasim, Modelsim or other HDL simulation tool.

1.1 TASKS

To complete this lab you must successfully complete the following tasks:

1.1.1 REGISTER FILE DESIGN

1) Build an RTL model of the Register File (RF). The definition of the module is given in the file `RF.sv` and its block diagram is given in figure 2.1.

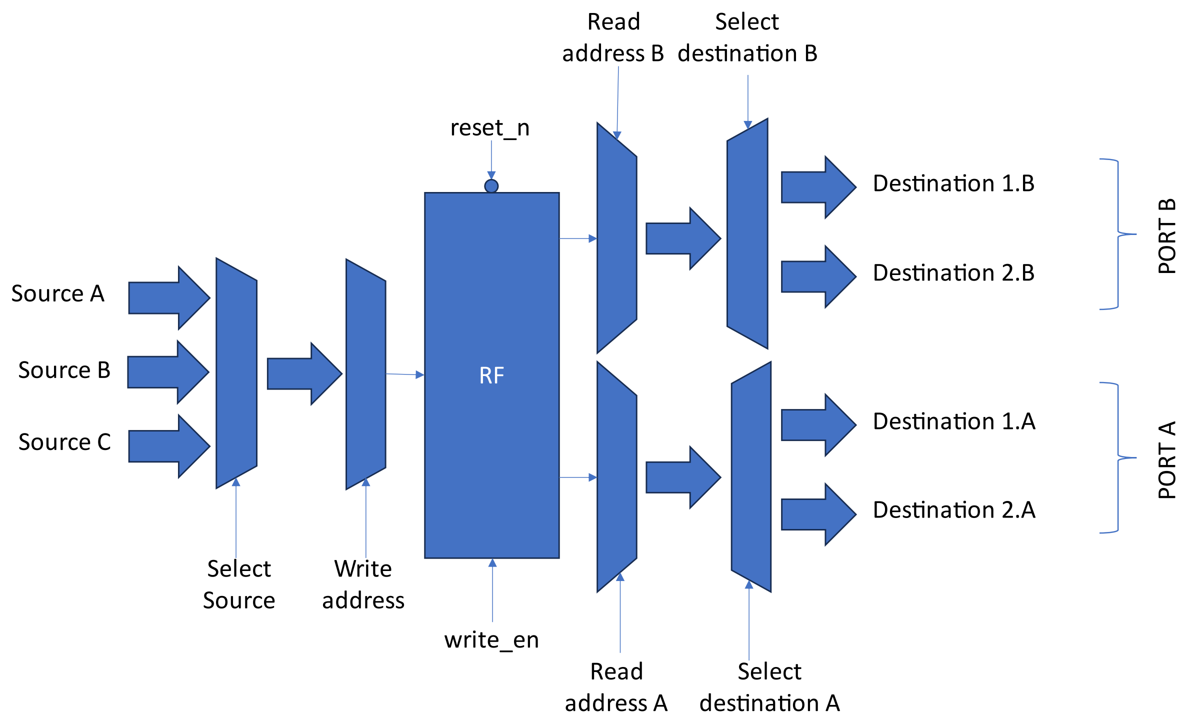

FIGURE 2.1 BLOCK DIAGRAM OF THE REGISTER FILE.

2) The RF should have 2 read ports and one write port, as seen in figure 2.1.

3) The write port should have a multiplexer that will select between three sources.

4) After the correct input is selected and the write enable (write_en) signal is asserted ('1') the data are written to the selected address.

5) Each read port receives a read address and the data from the read address is read out.

6) After the data from one of the port is read out, they fan out to one of two destinations. A select signal is used to select to which destination the data should propagate to, while the other is set to 0.

7) The design should be parametric. The source A, B, and C inputs as well as the Destination 1A, 1B, 2A, and 2B should be n-bit inputs and outputs respectively.

8) The read and write addresses should also be parametric a-bit inputs.

9) The select signals should be 1-bit for the destination, and 2 bit for the source.

10) The timing diagram shown in figure 2.2, describes the behaviour of the RF.

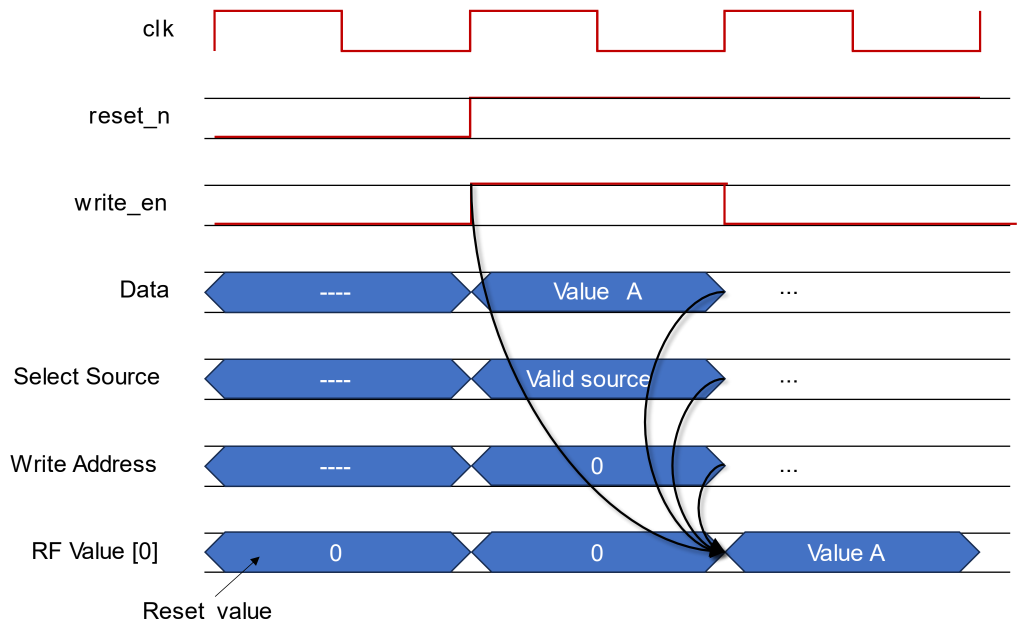

FIGURE 2.2A. WRITE TIMING DIAGRAM OF RF

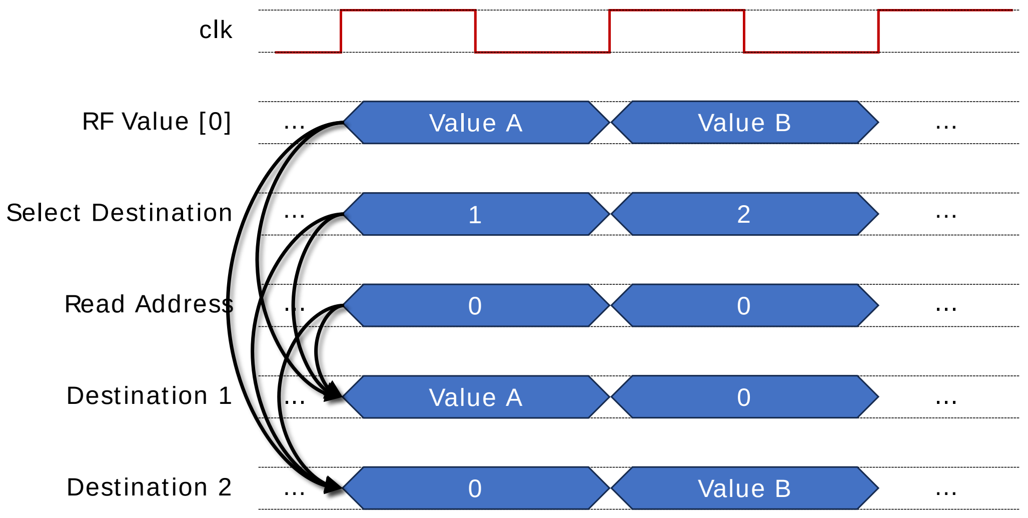

FIGURE 1.2B. READ TIMING DIAGRAM OF RF

11) The register file should use an asynchronous reset that sets the initial values of each word in the RF. That is, first location to signed 0, second location to signed 1, and all others to 0.

12) The first location of the RF (address 0) should be set to zero.

13) The second location of the RF (address 1) should be set to all ones.

1.1.2 ALU UPDATED DESIGN

Update the ALU that you designed in the previous lab to have register outputs as seen in figure 2.3.

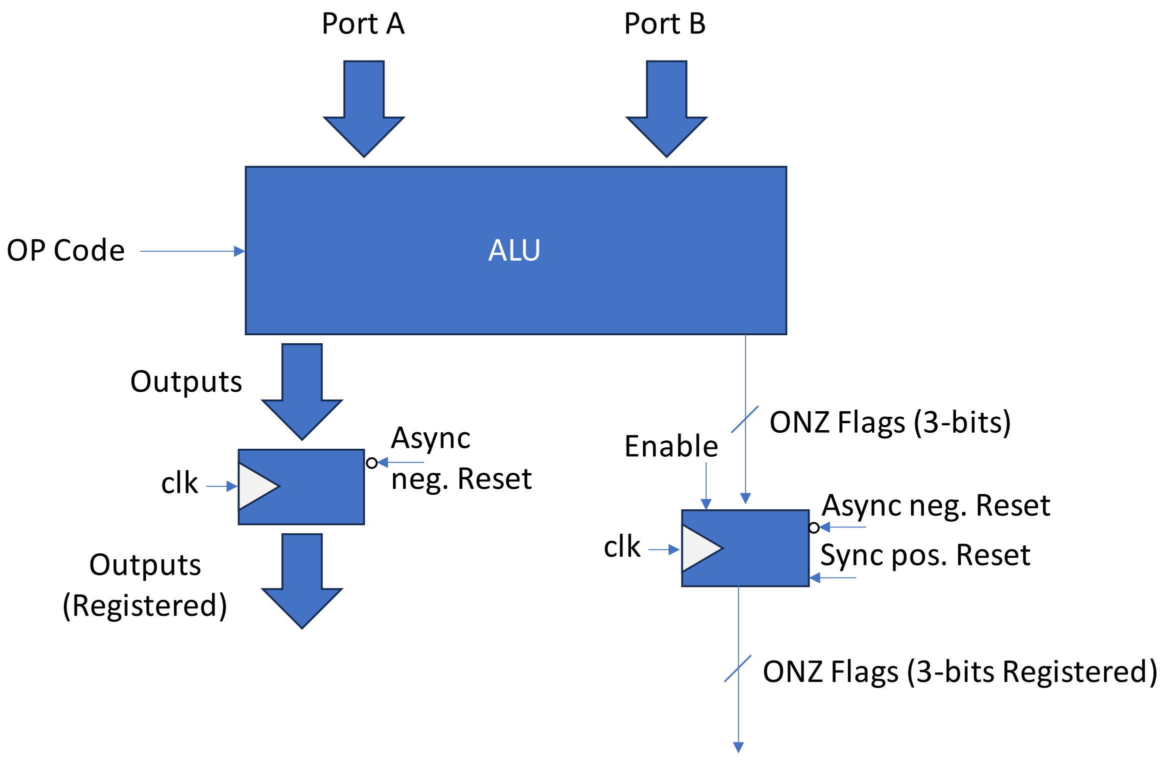

FIGURE 2.3 BLOCK DIAGRAM FOR THE UPDATED ALU

1) All ALU outputs are registered. All registers are using a asynchronous negative reset signal (i.e. the reset is not synchronized with the clock and it is asserted when 0). When the reset is asserted the registers should be set to 0.

2) The ALU ONZ flag register also has a synchronous positive reset signal, that can be used to clear it.

3) The ALU ONZ flag register should also have a synchronous enable signal that will only allow new data to be registered in when it is asserted to logic value ‘1’.

1.2 TESTBENCH REQUIREMENTS

Design the following testbenches to verify your register file and ALU designs.

1.2.1 RF TESTBENCH

Build a testbench that verifies the functionality of the RF.

1) The testbench has to check the reset, read and write functionality of your design.

2) The above check should be done for all addresses in the RF.

3) Make sure that you test all destinations and sources.

4) Create a scoreboard to make sure that all addresses has pass the check for reset, write and read.

Use random testing for your data and address inputs and directed testing for your select signals.

A scoreboard is a construct that counts the amount of correct and incorrect responses (outputs) from the design under test (DUT).

!Hint! You need to keep track of what is stored in your register file!

1.2.2 ALU TESTBENCH UPDATE

Update the testbench that you designed for your ALU to test the registered inputs and outputs.

1) Make sure that you adjust the timing of your testbench to take into account the registered outputs
2) Make sure that the control signals (reset and enable signals) of the registers operate as it should using directed testing.

1.3 DELIVERABLES:

1) All your files describing the register file and updated ALU design

2) All your files that implement your testbench