





Lab 7 – Register File

Objectives

In this lab, we will develop a Register File for a processor, and will simulate its behavior.

Section	
a) <u>Introduction</u>  A brief overview of a Register File.	05
b) <u>Implementation</u>  In this section, you will implement the Register File module.	60
<u>Exercise</u>  Small exercise to practice further.	60





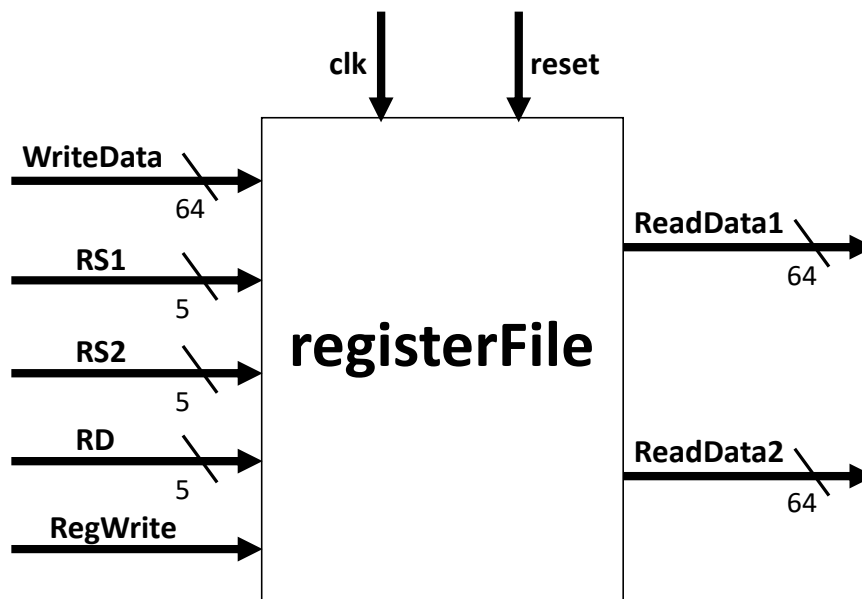
a. Introduction

Register files are necessary for computer memory. For a computer to be able to function, it needs to have some form of memory. There are two different forms of memory devices: external memory devices and local memory devices. External memory devices are items such as RAM, ROM, disk drives, etc. Registers provide local memory, which allows for temporary storage of data that is about to be processed. For this lab, registers will be used to save and store numbers.

b. Implementation

Create a module named, *registerFile*, which should have input address ports for reading register 1, register 2, and for writing data in a destination register. Furthermore, it should have a 64-bits data input port, and two 64-bits data output ports to read the values from two different registers. Finally, it should have a *clk*, *reset*, and a *RegWrite* signal which controls when the data should be written to a register. You need to define 32 64-bit internal registers.

The top-level diagram of this module is shown in the figure below.



As an example, the following command creates an Array of type `reg` containing 10 elements, each of which are 5 bits wide.



```
reg [4:0] Array [9:0]
```

where `Array` is the name of a variable. In your case, it should be `Registers`.



c. Design Requirements

- Initialize `Registers` with random values.



Use `initial` block to initialize the `Registers` with any random value.



To verify correct functioning of your design using test bench, make sure that different values are initialized in different `Registers`.

- The operation of writing data into a `Registers` should always be done when there is a positive edge of `clk` and `RegWrite` signal is asserted (or set i.e. HIGH).
- On `reset`, place a value 0 at both `ReadData` output ports.
- Reading a data from the register file should be made independent of `clk` signal. Reading should rather be sensitive to the change in inputs `RS1`, `RS2`, or `reset` or `Registers`.

Create a test bench to verify the correct functioning of designed module.



Exercise

Create a top module having 32-bit input, named `instruction`, and two 64-bit outputs, named `ReadData1` and `ReadData2`. Now, instantiate `instruction parser` module developed in lab 5 and `registerFile` module developed in this lab. Make appropriate connections.

Create a testbench and pass any 32-bit `instruction` in top module, and observe if the data of correct source registers are obtained.

Assessment Rubric
Computer Architecture Lab
Lab 07
Register File

Name:	Student ID:
--------------	--------------------

Points Distribution

Task No.	LR 2 Code	LR 5 Results	AR 7 Report Submission
Task 1 Register File	/30	/20	/20
Task 2 Exercise	/20	/10	
Total Points	/100 Points		
CLO Mapped	CLO 1		

For description of different levels of the mapped rubrics, please refer to the provided Lab Evaluation Assessment Rubrics.

#	Assessment Elements	Level 1: Unsatisfactory Points 0-1	Level 2: Developing Points 2	Level 3: Good Points 3	Level 4: Exemplary Points 4
LR2	Program/Code/ Simulation Model/ Network Model	Program/code/simulation model/network model does not implement the required functionality and has several errors. The student is not able to utilize even the basic tools of the software.	Program/code/simulation model/network model has some errors and does not produce completely accurate results. Student has limited command on the basic tools of the software.	Program/code/simulation model/network model gives correct output but not efficiently implemented or implemented by computationally complex routine.	Program/code/simulation /network model is efficiently implemented and gives correct output. Student has full command on the basic tools of the software.
LR5	Results & Plots	Figures/ graphs / tables are not developed or are poorly constructed with erroneous results. Titles, captions, units are not mentioned. Data is presented in an obscure manner.	Figures, graphs and tables are drawn but contain errors. Titles, captions, units are not accurate. Data presentation is not too clear.	All figures, graphs, tables are correctly drawn but contain minor errors or some of the details are missing.	Figures / graphs / tables are correctly drawn and appropriate titles/captions and proper units are mentioned. Data presentation is systematic.
AR7	Report Content/Code Comments	Most of the questions are not answered / figures are not labelled/ titles are not mentioned / units are not mentioned. No comments are present in the code.	Some of the questions are answered, figures are labelled, titles are mentioned and units are mentioned. Few comments are stated in the code.	Majority of the questions are answered, figures are labelled, titles are mentioned and units are mentioned. Comments are stated in the code.	All the questions are answered, figures are labelled, titles are mentioned and units are properly mentioned. Proper comments are stated in the code.