Temple University

College of Engineering

ECE 4612: Advanced Processor Systems

A red square with white letters

Description automatically generated

Adam Love

Register File Design

11/16/2023

**Objective**

This report details the design of the register file module. This module contains 32 64-bit registers. It allows 2 registers to output data at the same time, and allows 1 register at a time to have data written to it.

**Tools/Equipment**

This module and the module test bench were coded in Verilog using VScode. The module was tested using Vivado to generate output waveforms.

**Procedure**

To begin, it was known that the register file needs 32 total registers of 64-bits each. Because 2 registers can output data at the same time, 2 sets of 5-bit select inputs were needed to choose which register would output data. In addition, another 5-bit select input was needed to choose which register would have data written to it. There would need to be a 64-bit input to determine which data would get written to the selected write register. In addition, 2 more 1-bit input signals were needed to determine when to write to a specified register. These signals were the write signal (coming from the control module), and the clock signal. It should be noted that the register is written to on the negative edge of the clock signal. A zero register was also implemented. This register stays at value 0 even if it is attempted to be written to.

**Testing**

To test the register file, a test bench was generated that would attempt to write data to various registers, before checking what value is stored in the register. It attempted to write to registers while the write signal was disable, enabled, and also attempted to write to the zero register.

**Results/Observations**

A screen shot of a computer

Description automatically generated

The register file behaves as expected.

**Conclusion**

Initially I was trying to create discrete registers before implementing 32 total registers within the register file module. I realized that was the incorrect way of building this module very quickly. Building this module was very helpful to remind me of how Verilog allows an array of multi-bit regs to be created. Remembering that allowed this module to be created relatively pain-free.