# Assignment 3 - Registers, Data Memory, and Instruction Memory

**COEN 122L - Fall 2018**
**By:** Ryan Khodi (rkhodi@scu.edu)
**TA Office Hours:** By appointment.
**Professor:** Dr. Weijia Shang (wshang@scu.edu)

## Description

In this assignment, you will create the three main memory modules. The register file will contain all of the register memory for the system. The data memory component will be in charge of loading and storing the data. The last component is the instruction memory, which is in charge of holding the binary instructions for the system to run.

## Assignment

All three of the modules should be written in Verilog. When creating each of these components, we want to sync them all up to a clock. The purpose of the clock is to provide an alternating signal from high and low to ensure that each of these components act at the same time, this is referred to as sequential logic. In Verilog, this change is referred to as the posedge (positive edge or the change from low to high), or the negedge (negative edge or the change from high to low). For the purposes of this lab, use posedge.

In order to test your components, it is okay to use one test-bench to test all of your modules together.

### Register File

Our register file should contain 64 available 32-bit registers. The inputs of our register file are as follows: clock, write signal, three 6-bit addresses, and one 32-bit data in. The outputs of the register file should contain two 32-bit outputs (rs&rt) respectively.

### Data Memory

For our system, we are going to use 16-bit addressing. The reason for doing so is because if we use 32-bit addressing, the simulation software will most likely crash because that is a lot of addresses for the simulator to handle. Therefore, by limiting the range of addresses to 0-65536, the software will be able to run. In order to to access 16 out of the 32-bit address input, use 'address[15:0]'. The inputs of the data memory include: clock, write signal, read signal, 32-bit address, and 32-bit write data input. This component only has one output, and that is a 32-bit data out.

### Instruction Memory

The instruction memory module should be able to hold 32-bit instructions. Your module can hold as many instructions as you would like, but I suggest using 256. The inputs include: clock,

and a 8-bit address. The output is a single 32-bit instruction.

## Deliverables

To receive full credit, you will need to demo your working code. In addition, you must submit your source code (commented), your test-bench code (commented), and a screenshot of your waveform. To submit online, make sure everything is in a zipped folder (name the folder firstname_lastname.zip) and turn it into Camino. Please copy your code into individual .txt files and include those in the folder.

*This assignment was derived from Nate Matsunaga's*