# CSCE 230 Project

# Group #

Members:Johnathan Carlson,Tyler Zinsmnaster,Jake Ediger

Outline: This report will detail the following topics from Phase II:

1. Current Capabilities of Processor
2. Current Components/working parts
3. Process of connecting, implementing, and testing the various parts/functions

**1) Current Capabilities of Processor:**

The current capabilities of our processor are limited to say the least. With many stages still remaining, our processor is mostly reduced to the ALU (Arithmetic Logic Unit), the CU(Control Unit). We also created an assembler, although this was not part of phase II. The main function of the current state of the Processor is to execute R-type instructions. These instructions are relatively basic in their functions, although the implementation was not so simple. R-type instructions deal with registers, and what they can functionally do. R-type instructions can change based on the type of processor you are using. We used 24 bit instructions, which are designed to be compatible with our 16 bit FPGA’s.

**2) Current Components and Working parts**

We have several different components in our processor as of now. We designed the register file, ALU, and control unit. We were given as part of the project the Instruction Address generator, I/O memory interface, Memory interface, and Immediate block. The current working parts include the register file, the ALU, the control Unit, and partially working I/O.

**3)** Process of connecting, implementing, and testing the various parts/functions

To bring all of the components together, we used the programming language V.H.D.L., which stands for “Very High Speed Integrated Circuit Hardware Description Language”. It is a language developed for implementing logic onto a circuit, which we use to design our processor. Each of the parts of our processor are stored in different files in one overarching project. Similar to using functions in a language like C, we can use different components in different parts of the program. After you all the parts are connected(via PORT MAP), we can begin implementing the various components. In order to implement or test anything, it is necessary to first compile the program, and resolve any errors that may appear because of bad logic or syntax errors. After everything is resolved, the program can then be tested to see if any problems remain.

Testing is done by using a .do file to test certain(if not all) possible values, and then see if the output matches what is expected. “ModelSim” is the environment we use to simulate inputs and outputs. We added many different “waves” in the do file to check everything in modelSim to see what everything was doing at any given point. Also hand checked the values the assembler outputted to make sure it was operating correctly.