**EE 480 Assignment 2: Multi-Cycle Gr8BOnd**

Implementor’s Notes

Julianne Koenig

Dylan Etris

Michael Kamb

Department of Computer Engineering

University of Kentucky, Lexington, KY USA

**Abstract**

The project was a multi-cycle implementation of the instruction set design Gr8BOnd.

1. **General Approach**

The assignment was to design a Verilog multi-cycle implementation of the Gr8BOnd instruction set but not yet implement posit arithmetic. These instructions were treated as their equivalent integer operations. We also implemented an ALU to handle all arithmetic, logic, and integer-posit instructions.

1. **Issues**

Labels did not initially work as intended while testing our processor using assembly language. In place of labels, we used the actual addresses for branches and jumps. This problem was solved by incrementing the program counter after the end of each instruction, instead of in the start phase.

Setting the posit arithmetic functions to trap inside the ALU proved to be a challenge, as we could not figure out how to implement a trap signal wire to alter the current state. Instead, we chose to implement the posit instructions as integer instructions.

It was challenging to figure out how to handle a variable opcode field and feed the opcode to the ALU, but this was solved by setting the state to the first 4 bits of the opcode, and assigning an op register to the full 8 bit field. If the ALU was required for an instruction, the 8 bit opcode was needed for a case statement.

We were unable to resolve an issue where we got warnings compiling our code referencing the $readmemh functions.

We are unsure if the coverage analysis is useful, as our coverage analyses gave random values when performing the same test.