This semester I have been working with William Mansky on an end-to-end compiler verification project. In order to make my own contributions to the project, I first had to learn the basic functions of a compiler. At the highest level, the compiler translates a program written in a source language into a target language. Commonly, the source language is a high level language, such as C, while the target language is a low level language, such as machine code. The project I have been working on concerns what happens in the middle of the compiling process, before the target language code is generated. The compiler's first step is to translates the source language into an intermediate representation. A common language for the intermediate representation is the LLVM language which has specific features that make it desirable when performing compiler transformations. Once the compiler has generated the intermediate representation of the program, it will then attempt to optimize the program by making changes to it. These changes allow the program to use less system resources when executed, but the optimized program remains functionally equivalent to the original program that was written in the source language.

The purpose of the project is to validate the compiler's optimizations in order to ensure that the new optimized intermediate representation is functionally equivalent to the original. This is accomplished with the aide of a generic proof assistant program, Isabelle. A proof assistant is useful in making rigorous mathematical proofs because it is able to check that the proof is accurate. We use it not just to test, but prove that the optimized intermediate representations are functionally equivalent to the original. Isabelle accomplishes this by analyzing mathematical models of the compiled program's functionality. The models are developed by writing Isabelle programs describing LLVM language semantics.

These past few weeks I have mostly been working with my mentor to create these mathematical models in Isabelle by defining the operational semantics of LLVM instructions. For example, we
started with the instructions store and load, for storing and loading from memory respectively. First, I learned what operational semantics were, including small-step and big-step operational semantics, and how they represent the behavior of a program by defining valid transitions between instructions. I then learned how to write the semantics inside an Isabelle program that models the LLVM language. Each instruction is defined by rules for transitions on either instruction depending on current the state of the program and the arguments to the instruction. Next, we worked on adding support for the branch instruction. Previously, we had modeled the LLVM program as a list of instructions, where the next instruction executed is just the next instruction in the list. However, with the addition of branching instructions this model is no longer sufficient because branches allow the next instruction to be located anywhere in the program. Our solution was to create a graph model of the program, where each instruction is a node in the graph and the program is executed by following edges between different nodes. This improvement allowed for the addition of call and return instructions. The call instruction's semantics model an actual call instruction's behavior. It places the current program state or stack frame into a list of stack frames and executes the function that was called with a new stack frame. At the end of the function is a return instruction, which restores the previous stack frame from the list of frames and then executes the instruction following the original call instruction.

For the remainder of the semester I plan to continue to define more LLVM semantics in Isabelle, specifically the instructions necessary for optimizing concurrent code. This could involve making a more detailed memory model of the program's function as well as learning about what types of compiler optimizations are possible. I have had a very positive experience participating in the research being conducted at the University of Illinois. I found the work interesting and it has been very exciting to be working on projects that haven't been done before and have the potential to make a significant impact on the development of new technologies. Graduate school is certainly an option in the future, but before then I would like to gain more experience doing research during the Summer semesters in order to get a better idea of which engineering field interests me most.