Milestone #4: Code Generation

Overview

For this milestone, you will modify your compiler to generate assembly instructions as a first step toward a final translation.

Instruction Selection

Your program must generate ARM assembly for each input program, but without register allocation. Your program can accomplish this by translating from its internal LLVM representation into actual assembly, generating appropriate stack management instructions, and computing (initial) stack offsets.

This translation should support both the stack-based and the register-based (if implemented) versions of your LLVM intermediate representation.

Out-of-SSA

Note that the translation from LLVM to assembly must include translating "out-of-SSA" by replacing the phi instructions with an appropriate set of copy operations. The most direct approach is to create a new register for each phi, place in each predecessor a copy into that new register from the corresponding operand, and then replace the phi with a copy from the new register in to the target of the original phi instruction.

Output

Your program should default to printing the assembly to a file with a `.s` extension. Since register allocation is not yet implemented, you may verify that your assembly is correct for very small programs by "allocating" manually (for a small function, this is not as tedious as it may sound).

Test Machines

The department has a set of ARM servers that you can (eventually) use to test the output of your compiler (though you are unlikely to want to run your compiler on the servers, since they are rather slow). You can access these servers via the following host names.

```
pi01.csc.calpoly.edu
pi02.csc.calpoly.edu
pi03.csc.calpoly.edu
pi04.csc.calpoly.edu
```

Command-line Options

The output of your compiler should default to generating assembly code based on the LLVM representation with local variables in virtual registers. You should also add a `-llvm` command-line option so that you can view the LLVM representation.

Your compiler should support command-line option(s) to dictate which representation is emitted (e.g., without options, default assembly output based on virtual register-based LLVM; `-stack`, assembly output based on the stack-based LLVM; `-llvm`, llvm output based on the register-based LLVM; `-llvm -stack`, llvm output based on the stack-based LLVM).