THE UNIVERSITY OF ARIZONA.
DEPARTMENT OF COMPUTER SCIENCE

CSc 553: A Three-Address Intermediate Code Instruction Set for C--

The syntax of a three-address intermediate code is suggested below; the document Notes on Translating Three-Address Code to MIPS Assembly Code is based on this instruction set. While you are not required to use this particular intermediate code instruction set for your project, if you choose to use something different you will have to figure out the translation to MIPS assembly code yourself. (This usually isn't difficult to do, but something to be aware of nonetheless.)

Here, *n* denotes a (decimal) integer constant, and *label* denotes a label.

Data Declarations

global *id type*
: Declares *id* to be a global variable of type *type*, where *type* is **int**, **char**, **array[*n*] of int**, or **array[*n*] of char**.

Assignment Statements

x := y *op* z
: The usual binary operations: ***op*** is one of **+**, **-**, *****, **/**, and **y** and **z** are constants or variables.

x := -y
: Unary minus. **y** is a constant or variable.

x := y
: A copy operation. **y** is a constant or variable.

Jump Instructions

goto *label*
: Unconditional jump

if *x relop val* goto *label*
: *val* is an identifier or a decimal integer, ***relop*** is one of **<=**, **<**, **>=**, **>**, **==**, **!=**.

label *L*
: *L* is a label and must be unique in the program.

Indexed Assignments

x := y[i]
x[i] := y

Procedural Instructions

enter *f*
: *f* is (a pointer to the symbol table entry of) a function.

leave *f*
: *f* is (a pointer to the symbol table entry of) a function.

param *x*
: ***x*** is an actual parameter.

call p, *n*
: **p** is a procedure, *n* the number of arguments.

return
: return to the caller.

return *x*
: return to the caller, with return value ***x***.

retrieve *x*
: Copy the return value into ***x***.