Embedded Systems Engineering
Lab - 2-2: Clang AST Overview

R00159222 - Zachary Dair
zachary.dair@mycit.ie

This document provides a brief overview of the clang generated abstract syntax tree, for the while loop found inside the t2.c file.

We are examining lines 25-53 in the t2.ast file, which corresponds to a dump of the while loop.

This section starts with “WhileStmt” which corresponds to the call of the while statement, the following line “<<<NULL>>>” this may refer to additional arguments that can be passed to the while loop.

Following this we see the “BinaryOperator” class reference, of type int and value ‘<’

The “ImplicitCastExpr” class reference is then used to explicitly represent implicit type conversions, in this case to an ‘int’, then the “DeclRefExpr” is used to reference a declared variable, in this case the variable ‘i’. We then see the “IntegerLiteral” class reference, with the class ‘int’ and value ‘10’

From the “BinaryOperator” call and the following casts and references we can see this makes up the logic behind the “i less than 10” condition in the while loop.

We then see “CompoundStmt” which refers to the ‘{’, this represents multiple statements.

Following this is the “IfStmt” which corresponds to the if statement.

Similarly to the “WhileStmt” we see “<<<NULL>>>” for additional arguments.

Then the “BinaryOperator” class reference, of type int and value ‘&’, this operator is used for binary anding.

As in the “WhileStmt” we see an “implicitCastExpr” to an int, and a “DeclRefExpr” which is referencing the i variable, we can also see an “IntegerLiteral” for value 1, this means that the binary operator has been defined for the comparison, in this case binary anding, and the two values are retrieved by referencing the variable i, and by using an integer literal of value 1.

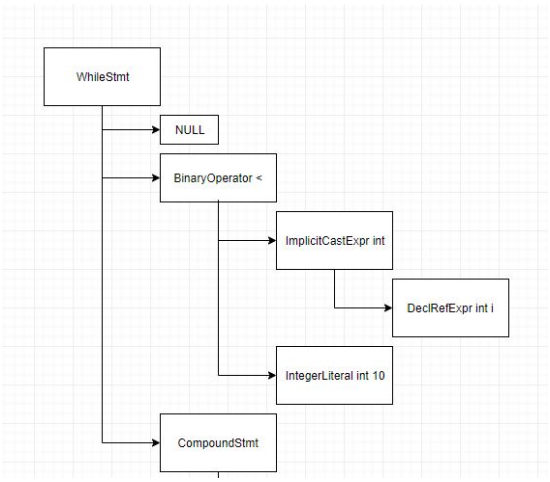
We then see a new “BinaryOperator” for value ‘=’, inside this we see the “DeclRefExpr” references j, and two further “BinaryOperators” using the ‘+’ as their operators. We can then see an ImplicitCaseExpr explicit casting to an int, for the “DeclRefExpr” referenced j variable, and similarly, the i value is referenced and cast to an int.

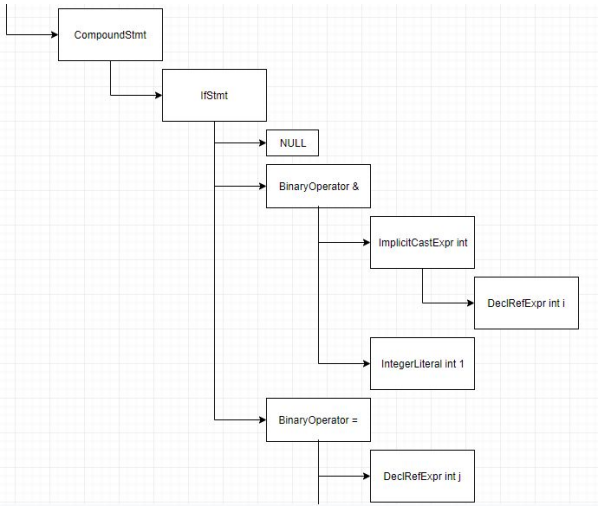
Using the first “BinaryOperator”, ‘+’ the j and i variables are added, and using the second “BinaryOperator”, ‘+’ and an “IntegerLiteral” 10 is added.

There is “<<<NULL>>>” found after these “BinaryOperator” lines, we can also see that this is the end of the “IfStmt”.

We can also see our final statement in the “CompoundStmt” sections, this is once again a “BinaryOperator”, ‘=’ for assignment, we then using “DeclRefExpr”, reference the i variable, using a subsequent “BinaryOperator”, ‘+’ add the values i and an “integerLiteral”, 1. This is done by using “ImplicitCastExpr” to explicitly case i to an int, and i is found using a “DeclRefExpr”. This means that i is referenced, and one is added to the variable i.

We can finally see the resolvement of the “CompoundStmt”, and the while loop.





Above are two portions of the abstract syntax tree in a diagram form.