CBMC is a Bounded Model Checker for C and C++ programs. It supports C version variants C89, C99, most of C11 and most compiler extensions provided by gcc and Visual Studio. A variant of CBMC that analyses Java bytecode is available as JBMC.

CBMC verifies memory safety (which includes array out of bounds checks and checks for the safe use of pointers), checks for exceptions, checks for various variants of undefined behaviour, and user-specified assertions. Furthermore, it can check C and C++ for consistency with other languages, such as Verilog. The verification is performed by unwinding the loops in the program and passing the resulting equation to a decision procedure.

Here, in our Project we use CBMC to generate the condition coverage of the code.

Let’s Discuss about **CONDITION** **COVERAGE.**

Condition Coverage or expression coverage is a testing method used to test and evaluate the variables or sub-expressions in the conditional statement. The goal of condition coverage is to check individual outcomes for each logical condition.

Condition Coverage is a level of test coverage that requires every combination of the outcomes of sub-conditions within a compound condition to be tested. For example, a compound condition might take the form of ((a == b) and (x < y)). Multiple condition coverage would require four tests—one where both (a == b) and (x < y) are true, one where (a == b) is true and (x < y) is false, and so on.

Condition coverage offers better sensitivity to the control flow than decision coverage. In this coverage, expressions with logical operands are only considered.

For example, if an expression has Boolean operations like AND, OR, XOR, which indicates total possibilities.

The formula to calculate Condition Coverage:

