Document: WG14 N1353

Submitter: Fred Tydeman and Jim Thomas (USA)
Submission Date: 2009-02-26
Related WG14 documents: N1321, DR 290
Subject: FLT_EVAL_METHOD issues

This paper notes problems related to the following sections of text, in C1x draft N1336, as recently modified by N1321, and proposes ways to fix them.

5.2.4.2.2 #8

Except for assignment and cast (which remove all extra range and precision), the values of operations with floating operands and values subject to the usual arithmetic conversions and of floating constants are evaluated to a format whose range and precision may be greater than required by the type. The use of evaluation formats is characterized by the implementation-defined value of **`FLT_EVAL_METHOD`**:20)

6.8.6.4 #3

If a **`return`** statement with an expression is executed, the value of the expression is returned to the caller as the value of the function call expression. If the expression has a type different from the return type of the function in which it appears, the value is converted as if by assignment to an object having the return type of the function.142)

142) The **`return`** statement is not an assignment. The overlap restriction of subclause 6.5.16.1 does not apply to the case of function return. The representation of floating-point values may have wider range or precision and is determined by **`FLT_EVAL_METHOD`**. A cast may be used to remove this extra range and precision.

Problem 1: The term “operations” is potentially misleading. The standard uses “operations” as a more general term than “operators”, for example, in “Operations on files”. One might wonder if library functions like sqrt() could be affected by wide evaluation. But only operators are subject to the usual arithmetic conversions, so “operators” could replace “operations” in the text to clarify the matter, without substantive change.

Recommended change:

5.2.4.2.2 [8]: change “operations” to “operators”.

Problem 2: The new text in 5.2.4.2.2 and footnote 142 now implies that function returns are widened by widening expression evaluation methods. This specification is not tenable from an ABI perspective because it would introduce inconsistencies between the format expected by the callers and the format returned by the callee, depending on the evaluation method used for their translation. The following recommended changes revert to the original intention, namely that function returns, like assignments and casts, are not affected by wide expression evaluation methods.

Recommended changes:

5.2.4.2.2 [8]: change “Except for assignment and cast” to “Except for assignment, cast, and return”.

6.8.6.4 #3: change the second sentence to “If the expression is evaluated to a format different from the return type of the function in which it appears …”

Footnote 142: remove the last two sentences (thereby reverting to the C99 text).