Document: N1730
Date: 2013/07/21

Floating-point issues in C11 from PDTS 18661-1 UK review

Joseph Myers

Some issues with floating point in C11 have been identified as part of the UK review of the N1711 draft of TS 18661-1. While such issues relate to the general area of C bindings to IEC 60559:2011, and so could be addressed in the TS on that basis, since the issues also apply to C11 as-is it may be more appropriate to address some or all of these issues as Defect Reports with a view to having a normative fix in a future TC to C11 rather than only having a fix in conjunction with the new bindings.

Issue 1: Choice of long double in Annex F

Annex F provides various choices for the long double type (which may or may not be an IEC 60559 type), but no method is provided for an application to determine which choice has been made.

If a macro is provided to say whether the type is an IEC 60559 type, all the other properties can be determined from the existing <float.h> macros. So, a sufficient fix would be:

In 5.2.4.2.2, insert a new paragraph after paragraph 10: Whether a type matches an IEC 60559 type is characterized by the implementation-defined values of FLT_IS_IEC_60559, DBL_IS_IEC_60559, and LDBL_IS_IEC_60559:

```
  0 type does not match an IEC 60559 format

  1 type's values and operations are those of an IEC 60559 basic,
    interchange or extended type
```

Issue 2: Definition of FLT_ROUNDS

The C11 definition of FLT_ROUNDS is inadequate in that it refers to floating-point addition but does not say addition of what type. If long double is not an IEC 60559 type, it may not fully support all rounding modes even though they are supported by other types. (This is an actual issue with real implementations using non-IEC 60559 types for long double.) A suitable fix would be:

In 5.2.4.2.2#8, insert "for type float" after "floating-point addition". At the end of F.2#1, insert "The value of FLT_ROUNDS applies to all IEC 60559 types supported by the implementation, but may not apply to non-IEC 60559 types.".

Issue 3: Floating-point exceptions and 6.5#5

C11 6.5#5 says "If an exceptional condition occurs during the evaluation of an expression (that is, if the result is not mathematically defined or not in the range of representable values for its type), the behavior is undefined.". When the Annex F bindings are in effect, it must be intended that floating-point exceptions do not produce such undefined behavior (instead, behavior such as evaluating to a NaN must be defined). But no normative text actually says that. A suitable fix would be:

Append to 6.5#5: For implementations defining __STDC_IEC_559__, this does not apply to exceptional conditions where the behavior (such as raising a floating-point exception and returning a NaN) is defined by

Annex F, directly or by reference to IEC 60559.

Issue 4: Floating-point state not being an object

The description of the floating-point environment in C11 fails to make sufficiently clear what is or is not an object (C11 footnote 205 is not normative, and so cannot be used to that effect); it uses terms such as "system variable" without saying what that is. Simply moving that footnote to normative text would fix this issue:

Move the contents of footnote 205 (C11 subclause 7.6) to the end of 5.1.2.3#2.