WG14 N2910
Meeting notes

C Floating Point Study Group Teleconference

2021-12-22
8 AM PST / 11 AM EST / 4 PM UTC

1) Attendees:
Jim T, Mike C, David H, Ian M, Damian M

2) Added WG14 paper to agenda.

3) Rajan's notes from previous meeting posted.

4) Next CFP meeting 19 Jan 2022.

5) Next WG14 virtual meetings Jan 31-Feb 4 AND Feb 14 - 18 (all 2022).

6) CFP meeting to be held Feb 9 if needed in between WG14.

7) [2302] C++ liaison issue from David Olsen
Relates to arith conversions occurring between long double and _Float64.
David Olsen says an expression converting to long double is more consistent.
But, the current draft says that such an expression converts to _Float64.
It was noted that a standard type may have lesser quality arithmetic than
an interchange type which is at odds with the fact that we really want to
lean towards better quality arithmetic
Jim proposed some text if we decide to change.
It was also noted that such a change also
* affects tgmath rules for which examples need to be checked, and
* adds another special case to an already complicated set of rules.
Motivated by Microsoft Visual C++ where long double is the same as double.

Conclusion: Recommend no change - change does not justify added complexity.

8) Note that the current C2X draft is still n2596.pdf.

9) Jim noted [2260] Meaning of nearest in case of overflow - Lefevre.
"nearest" can never round to infinity
This is related to other issues raised by double-double.
See below.

10) Updates to accommodate double-double had further consequences.
Led to issues with HUGE_VAL in nextup/down specs.

11) N2842 - insert sentence allowing double-double to classify all full-precision representations as normal.

Jim’s edit accepted

AI: Jim Thomas submit to WG 14

12) update N2843/2596 - max exponent macros - to clarify double-double

Jim's edit accepted

AI: Jim Thomas submit to WG 14

13) 5.2.4.2.2 update N2806/2596 - diffs are given relative to current draft
and relative to last update. This includes definitions of normalized etc.

AI: Jim Thomas - copy changed footnote 22 into set of suggested changes to
N2806+N2596

Separate paragraphs make definitions explicit instead of parenthetic
remarks and seems better structurally.

Note that this might change the semantics relative to some people's
interpretation of the previous text.

AI: Jim Thomas - change the last item which says "are called zero floating-point
numbers" to just "are zeros".

We considered changing all the "are called" to "are" and decided not to.

AI: Jim Thomas submit to WG 14, after making changes above.

14) Overflow and underflow definitions changes to N2596+N2805.

a) 3.9 change note
says rounding is implementation defined if there's an infinity in the format

b) nextup/nextdown formerly referred to HUGE_VAL - which is no longer mentioned

AI: Jim Thomas change 7.12.11.5 nextup to match 11.6 nextdown as follows:

If x is the positive number (finite or infinite) of maximum magnitude
in the type, nextup(x) is x....

And submit to WG 14.

15) no action on HAS_SUBNORM

16) Jim wrote a note to editor who then responded that he was aware of the
numerous changes on math.h and float.h, coming from the many proposals.
There is a vast opportunity for confusion.

AI: Jim Thomas - track the many changes to infinity and nan macros.

17) 2256 - printf and rounding - raised by Vincent
What happens when you ask for more digits than are guaranteed to
be correctly rounded?
It Would be possible to ask for more digits and get less a accurate
result.
This is not likely to happen and is it worth fixing?
It is actually too late for a new proposal.

No action - this is a quality of implementation.

AI Fred: - add to list of issues to be revisited for next version.

18) N2726 imaginary i in complex - remove const qualification in macros

AI: all - review N2726 for next time.

19) 2258-2271 terminology

When does "floating-point number" refer to the model and when not?
We decided it was too big to tackle this issue this late in the game.

Damian was reminded that Infinity is NOT a floating point number.

fabs issue - floating-point number x - not mentioned in other lib functions.
This could be just an editorial to remove "a floating-point number" .

AI: Jim Thomas - send note to editor asking for editorial change in 7.12.7.~~2~~3.
Copy Rajan and Fred

20) It was noted that range is used in two different ways
* "range of representable values" is very specific, and
* "range" alone refers to exponent range.

Vincent wants clarification of different uses.

5.2.4.2.2 has already become almost unreviewable, because of pending changes.

AI: anybody interested in 5.2.4.2.2 - review issue and make recommendation.