WG14 N3100
Meeting notes

C Floating Point Study Group Teleconference

2023-01-31
8 AM PST / 10 PM EST / 4 PM UTC

Attendees: Rajan, Jim, Fred, Damian, Ian, Vivian

New agenda items (https://wiki.edg.com/pub/CFP/WebHome/CFP_continuation_meeting-20230131.pdf):
None.

Next Meeting(s):
February 8, 2023, 4PM UTC - If needed
February 28, 2023, 4PM UTC
ISO Zoom teleconference
Please notify the group if this time slot does not work.
Note: The WG14 meeting is February 13-17, 2023
Note: Cancelling February 14th, 2023 meeting due to the WG14 meeting being at the same time.

Carry over action items:
Skipped.

Last meeting action items:
Skipped.

New action items:
Jim: CFP response to US42-169 should be copy the nextafter returns section, changing "nextafter" to "nexttorward".
Jim: Add in a link to CFP 2657 into CFP's response for GB-287.
Jim: Look at FENV_ROUND and use similar words from FENV_DEC_ROUND (7.6.3#2) to show the distinction between constant rounding modes and dynamic rounding.
Jim: Add in something about assuming the decimal point is a single character to the comment for the #define MAXSIZE (or mention the C locale is assumed) in H.12.2#4 as a CD2 comment after clearing it with Vincent and an FYI to WG14.
Jim: Submit a CD comment resolution document for US42-169, GB-286, GB-287.

C++ liaison:
Skipped.

C23 integration:
CD1 ballot resolution first half meeting results being discussed this CFP meeting.

Carry-over action items results:

David H: Get an example for the scaled reduction functions (perhaps by asking Jason or Jim or looking into the IEEE references). - Not done.

See https://754r.ucbtest.org/background/traps-and-wraps.txt

David H: Get an example for the augmented arithmetic functions (perhaps by asking Jason or Jim or looking into the IEEE references). - Not done.

Action items results (from previous meeting):

Skipped.

Discussion:

WG14 meeting results:

N3067 2022/12/21 Keaton, CD 9899 ballot comments

N3082 2023/01/21 Thomas, CFP Review of NB comments - N3081 update

See [Cfp-interest 2639]

Issues with N3082:

US42-169

See [Cfp-interest 2640]

CFP answer to the comment is that yes there should be a Returns section. The reference was there to show for related functions, they all have a returns section.

^AI: Jim: CFP response should be copy the nextafter returns section, changing "nextafter" to "nexttorward".

GB-286

See [Cfp-interest 2655]

Jim: Missed the encoding functions. Added them as well.

Jim: The last change here depends on CB-287's (hex input for strtod*) resolution.

GB-287

See [Cfp-interest 2655, 2633, 2649, 2650, 2657, 2658, 2653, 2652, 2637]

Jim: This simplifies the specification. Makes it the same specification as the strto{f/d/ld}, etc. functions. The changes look big, but are moving around a lot of text. Hex can't always be correctly rounded so that part has to go. Otherwise the rest of the text moves to make it similar to strtod. The hex input part was taken from Annex H other than the * (for a footnote) marked sentence.

Fred: Did the (P-1)/4 come from 754 or did you figure it out yourself?

Jim: That was from 754.

Jim: The meta-question about whether this is needed or not is there. IEEE doesn't require hex input for decimal. But there is a requirement to be able to convert between all formats. This includes arithmetic and non-arithmetic. Including from binary to decimal. See H.12.2#4 for an example as to the only way to do (non-arithmetic)binary128->(arithmetic [though could be non-arithmetic for the argument of needing hex input])decimal128 since you need hex input for a single rounding meaning strtod128 needs to be able to handle hex.

Fred: Can you convert non-arithmetic binary to a decimal string?

Jim: Yes, you can use the strfromenc* functions.

Fred: That will have a rounding. Then do you have one from that string to a decimal type?

Jim: How do you know how many digits to carry, and knowing the rounding mode and hiding the inexact exceptions (if any).

Fred: I'll think about it.

Jim: Joseph has a good explanation of the issues of trying to do your (Fred's) way in CFP2657.

Rajan: We can put a link to that comment in our response doc to WG14.

^AI: Jim: Add in a link to CFP 2657 into CFP's response for GB-287.

Fred: Can you say any hex value can be given in decimal given enough decimal digits?

Jim: Yes.

Fred: Except there is no print function required to give that many digits.

Jim: Yes.

Fred: You'd need 16K digits. The smallest denorm would take the most digits.

Other issues:

Floating constants

See [Cfp-interest 2659]

Preference for the second wording of CFP2659.

Vincent said if it is FE_DYNAMIC it doesn't work.

Rajan: The part about other than "FE_DYNAMIC" should be interpreted for the last part of 7.6.2#4 as well. It is not close enough though so it can be interpreted the other way.

Fred: Can we add in "translation time"?

Jim: Floating constants are always evaluated at translation time. It is the assignments that are evaluated at runtime.

Rajan: Paragraph 3 in 7.6.2 does imply FE_DYNAMIC is a constant rounding mode. We need to define "constant rounding mode" to not include that.

Jim: We can change "the specified constant round mode" in that paragraph to "direction".

Jim: We should look to see where else we say FE_DYNAMIC is a rounding mode. It may need to be a CD2 comment.

Rajan: Paragraph 5 of 7.6.2 also implies FE_DYNAMIC is a constant rounding mode.

Rajan: We could do something similar to 7.6.3#2's end to show the difference for binary as well.

^AI: Jim: Look at FENV_ROUND and use similar words from FENV_DEC_ROUND (7.6.3#2) to show the distinction between constant rounding modes and dynamic rounding.

Suggestion: See if the changes work with Vincent first and, optionally bring it forward to WG14, prior to a CD2 comment.

Definition of “floating types”

See [Cep-interest 2654]

Rajan: As long as float_t and double_t have to be standard floating types, then we are good and there is no contradiction since real floating types contains standard floating types.

Fred: We should change float_t and double_t to be real floating types since right now they could be complex.

Needs to be looked at further.

Locale issue in example H.12.2p4

See CFP2665, 2666.

Jim: We could say this is for the "C" locale or something else appropriate. Ex. in H.12.2#4

Fred: Could also put it in the comment for MAXSIZE in the code as a parenthetical.

^AI: Jim: Add in something about assuming the decimal point is a single character to the comment for the #define MAXSIZE (or mention the C locale is assumed) in H.12.2#4 as a CD2 comment after clearing it with Vincent and an FYI to WG14.

^AI: Jim: Submit a CD comment resolution document for US42-169, GB-286, GB-287.

Rajan Bhakta
z/OS XL C/C++ Compiler Technical Architect
ISO C Standards Representative (Canada, USA), PL22.11 Chair
C/C++ Compiler Development