WG14 N2870
Meeting notes

C Floating Point Study Group Teleconference

2021-10-13
8 AM PDT / 11 AM EDT / 3 PM UTC

Attendees: Rajan, Jim, Fred, Damian, Mike, Ian, David O., David H.

New agenda items (https://wiki.edg.com/pub/CFP/WebHome/CFP_meeting_agenda_20211013-update.pdf):
None

Carry-over action items:
None

Last meeting action items (done unless specified otherwise, details below):
Jim: Propose new wording for N2716's change that WG14 did not accept as is.
Rajan: Get a document number for CFP2140 and send an updated document to CFP before WG14.
Jim: Put the formulas into CFP2130's text and show CFP before submitting it to WG14.
Jim: Reword the footnote for CFP2153 and show it to CFP before submitting it to WG14.
Jim: Send out a change to CFP changing the wording of the FLT_MAX_EXP family of macros to say they are for normalized numbers.
Anyone: Look into the use of the term "floating-point number".
Rajan: Get back to Aaron about the FLT_EVAL_METHOD constant value question and point him to part 5 of the TS which does have changing values.
All: Look at CFP2136 and respond within a week.
Jim: Submit a paper to remove the promotion rules as per discussion in CFP2141.
Jim: Propose the change in CFP2154 to WG14.

New action items:
Jim: Update the INFINITY macro paper to define INFINITY iff infinities exist in type float, and as an alternative, do what is in the current paper.
Rajan: Bring up CFP's position in WG14's liaison report on not proposing double and long double INFINITY macros despite it being brought up. If WG14 wants it, let us (CFP) know.
Jim: Change N2716's alternative wording proposal to replace the second "=" in the example with "yields" and the "page 450" with the section, sub-clause number.
Jim: Look at the changed text and why some unintentional underlining is present in C23_proposal_-_Normal_and_subnormal_classification-20211008.pdf (Ex. emax).

Next Meeting(s):
Note: New day for this meeting only.
Same time slot.
Tuesday, November 23rd, 2021, 4PM UTC
ISO Zoom teleconference
Please notify the group if this time slot does not work.

November 10th, 2021 reserved as a potential date as well at the same time in case we need it.

Upcoming WG14 meetings:

November 15-19, 2021, virtual, 14:30-18:00 UTC each day
Mailing deadline: October 15th, 2021 (Document number requests due October 8th, 2021)
January 31-February 4, 2022, Portland, Oregon, US (Tentative)
Mailing deadline: December 31, 2021
July 11-15, 2022, Strasbourg, France (Tentative)
Mailing deadline: June 10, 2022

C++ Liaison:

WG21's SC22 special meeting about C/C++ compatibility (See CFP2224):

David O: The topic of discussion was the C++ proposal and the interaction with C. Re the new annex. Mostly set. Nothing that C FP needs to follow up on. Naming with _F* names in C++ is still muddled. For arithmetic conversions, we want the interchange type vs the standard type. This is being changed now in the C++ proposal to match C. Implicit conversions difference is staying since it is compile time.

Rajan: My interpretation was the objection to the _Float* name is really just not liking the name, not a technical reason behind it.

David O: Yes, there was some confusion that was addressed. The objection to keywords was adding it to the grammar I believe, and having them optional is not currently previously available in C++. Also having it in the global namespace was an objection. C++ does not like typedefs in the global namespace.

Jim: I attempted to explain the reason and meaning of the _t C names. I don't think anything came of that. Let me know if you need help with this for C++.

David O: I believe there was a poll to change the _t names in C++ but it failed. I plan on keeping it with different meanings between the languages.

C23 integration:

Latest C2X drafts: http://www.open-std.org/jtc1/sc22/wg14/www/docs/n2596.pdf http://www.open-std.org/jtc1/sc22/wg14/www/docs/n2573.pdf http://www.open-std.org/jtc1/sc22/wg14/www/docs/n2478.pdf

IEC 60559:2020 support

Jim: Still no new draft.

Jim: In the future, we should just focus on what's wrong, vs what is unclear or new features. Is it a known problem for users or implementors? If not, it would be hard for us to consider it. Clear editorial corrections should be fine.

Jim: At some point, we'll have to review C23 with our changes.

Rajan: Also the TS updates rebased on C23.

Action item resolutions:

Jim: Propose new wording for N2716's change that WG14 did not accept as is (https://wiki.edg.com/pub/CFP/WebHome/C23_proposal_-_Revised_suggested_change_from_N2716-20211008.pdf):

Rajan: Like it, avoids the equal/numerically equal issues.

Jim: The second = in the example should be "yields".

David H: Can use a right arrow.

Jim: That works for transformations, and this is not.

All: "yields" seems good.

Jim: Change the page number to a subclass?

Ian: Agreed.

AI: Jim: Change N2716's alternative wording proposal to replace the second "=" in the example with "yields" and the "page 450" with the section, sub-clause number.

Rajan: Get a document number for CFP2140 and send an updated document to CFP before WG14 (See CFP2165, N2823):

Rajan: Sent out to WG14 as N2823.

Jim: Put the formulas into CFP2130's text and show CFP before submitting it to WG14 (https://wiki.edg.com/pub/CFP/WebHome/C23_proposal_-_Normal_and_subnormal_classification-20211008.pdf):

Rajan: See some unintentional underlining. Fred does as well.

Jim: Not intended. I will look at it.

AI: Jim: Look at the changed text and why some unintentional underlining is present in C23_proposal_-_Normal_and_subnormal_classification-20211008.pdf.

Jim: Reword the footnote for CFP2153 and show it to CFP before submitting it to WG14 (https://wiki.edg.com/pub/CFP/WebHome/C23_proposal_-_Clarification_about_expression_transformations-20211003.pdf):

Fred: Looks good.

Ian: The compilers I worked on at IBM were modified to have an option to have strict or not. Including sub-options to tune it.

Jim: Send out a change to CFP changing the wording of the FLT_MAX_EXP family of macros to say they are for normalized numbers (https://wiki.edg.com/pub/CFP/WebHome/C23_proposal_-_Clarification_for_max_exponent_macros-20211002.pdf):

Fred: Now you are mixing max float which does not have to be normalized.

Jim: We saw there is no consistency in the naming of these. We haven't chose to change anything for consistency.

Fred: It only matters for double double. I don't know if it has any numbers bigger than the maximum normalized.

Jim: I believe it does.

Fred: If both parts are double max, is that a valid number People said no.

Ian: Depends on the implementations, the original by IBM on AIX did, while the GCC on linux one didn't. GCC did a lot of instructions to make the first part be the same as a double which changed the semantics.

Jim: My sense is (examples not coming to mind right now) that examples are obvious if you don't require the form for double double, but also examples if you do, but more subtle.

Jim: The rationale paragraph right before the change should cover what Fred is talking about as to why.

Fred: For FLT_EPSILON we did change it in C99.

Jim: Unfortunately this was done one at a time. It wasn't done everywhere.

Fred: We'll see if anyone else objects. Joseph Myers may notice and object.

No objections to submit this.

Anyone: Look into the use of the term "floating-point number" (See Cfp-interest 2110):

Skipping.

Rajan: Get back to Aaron about the FLT_EVAL_METHOD constant value question and point him to part 5 of the TS which does have changing values (See CFP2166,2170):

Rajan: Aaron was happy with this.

Jim: Compiler magic can enable #if/#elif with FLT_EVAL_METHOD if they wanted to.

Rajan: True, but generally not talked about and doesn't work well with preprocessing separately.

All: Look at CFP2136 and respond within a week (ttps://wiki.edg.com/pub/CFP/WebHome/C23_proposal_-_Type_annex_tgmath.h_narrowing_macros_with_integer_args-20211008.pdf):

Looks good.

Jim: Submit a paper to remove the promotion rules as per discussion in CFP2141 (https://wiki.edg.com/pub/CFP/WebHome/C23_proposal_-_Remove_default_argument_promotions_for__FloatN_types-20211008.pdf):

Looks good.

Jim: Propose the change in CFP2154 to WG14 (https://wiki.edg.com/pub/CFP/WebHome/C23_proposal_-_feraiseexcept_update-20211010.pdf):

Looks good.

Other issues:

Contradiction about INFINITY macro (https://wiki.edg.com/pub/CFP/WebHome/C23_proposal_-_Contradiction_about_INFINITY_macro-20211010.pdf):

Jim: The iff INFINITY is present, define it, could follow something like the NAN macros. That would allow INFINITY to be a feature test macro. This would be a substantive changed. I could also break code that depended on UB in a certain way. Is what we have OK?

Fred: The character sequence could be 1E9999?

Jim: Yes.

Rajan: This disallows the "(const float)1E9999" for example.

Fred: You could add in a trailing 'f' to make it a literal.

Jim: I don't thing having the cast makes it a constant. It would not be allowed.

Fred: The existing wording, says INFINITY has to expand to a constant expression which allows the cast.

Damian: Are we allowing this to be an expression?

Jim: The problem is in the 'else' clause which doesn't have the INFINITY. My first attempt was to say "else to a constant expression".

Fred: Can make the second change into "constant expression"?

Jim: Makes it not read well in terms of "character sequence in the form of a constant expression". Not sure why my previous wording was changed (from else constant expression).

Rajan: The 1E999 is a double type which can hold the value, and when converted to a float, it is still a constant expression which had that value become an infinity.

Ian: Perhaps "the constant in the evaluation format of type float"? If the evaluation format is double, the evaluation format is double, of type float.

Jim: That is getting into the second problem. Lets try to solve the first one.

Fred: If you don't have INFINITY, then it won't work. If you do have an INFINITY you don't have an issue.

Jim: Yes, if you have an INFINITY, there is no issue.

Jim: A problem with what was proposed can't be done at translation time if FENV_ACCESS is on.

Jim: We could change it to make INFINITY defined iff INFINITY exists. Do we want a backup if it fails?

Fred: We could have a feature test macro to say if infinities exist.

Jim: INFINITY could become the feature test macro.

AI: Jim: Update the INFINITY macro paper to define INFINITY iff infinities exist in type float, and as an alternative, do what is in the current paper.

Ian: It does bring up the question if every floating point type should have INFINITY.

Damian: The float constant that is infinity, will always be infinity in double and long double.

Jim: This is true. But Ian brought up CELL which has no infinity for float, but it is present for double.

Jim: Should we define double infinity and long double infinity?

Ian: Is there any current development on Cell or changes coming?

Jim: I would suggest not proposing this. It would be a new feature. If WG14 wants this they could ask us.

AI: Rajan: Bring up CFP's position in WG14's liaison report on not proposing double and long double INFINITY macros despite it being brought up. If WG14 wants it, let us (CFP) know.

Wrapping up proposals (See CFP2203,2223):

Others?

David H: There will be a virtual ARITH conference and the next one will be live.