WG14 N2942
Meeting notes

C Floating Point Study Group Teleconference

2022-02-10
9 AM PST / 12 PM EST / 5 PM UTC

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

New agenda items (https://wiki.edg.com/pub/CFP/WebHome/CFP_meeting_agenda_20220210-update.pdf):
Paul Zimmerman's email CFP2370.
Fred: What about adding an 'i' suffix for imaginary (posted in WG14's reflector).

Next Meeting(s):
March 16th, 2022, 3PM UTC (8am Pacific time expected due to Daylight Saving Time in parts of USA)
ISO Zoom teleconference
Please notify the group if this time slot does not work.

Mike: Will be unavailable then. +/- a week each way as well.

WG14 meeting:
See CFP2366, CFP2361-CFP2363, CFP2367
N2860: No reservation of math function names as external identifiers. Not everyone has a separate math library.
David H: I agree that it is not good to split the C library.
Jim: My first reaction is negative with it. C code could have these identifiers mean anything on other implementations. It's selling problems.
Fred: A lot of implementations have libm. Jen's basic issue is the addition of so many identifiers. He doesn't like it. I believe Jens proposal to make true/false identifiers will break more things than all of the math stuff.
Jim: The identifiers in math.h are reserved right now externally?
Rajan: Yes. Still a portability concern if they use the name for something else. Ex. cos could be used for something else.
Damian: How many things is this going to break? Currently neutral.
Jim: About breaking more things in the future, Jens concern is more a constraint on programming due to reduced user namespace. I believe the only objections have been from Jens, none from users. There are implementations out there and no user issues.
Fred: Only POSIX and gamma had a namespace issue in the past which is why we came up with tgamma.
N2887: Example using sqrt as an unsequenced function.

C++ Liaison:
Nothing new.

C23 Integration:
Review draft: See CFP2320
Discussion under CFP2319 and follow ups.
Jim: The annex is worrisome since it is a big annex.

Carry-over action items:

Fred: Ask C++ what their issues with *_HAS_SUBNORM are and if they are OK with obsoleting it. - Still open

Fred: Their numeric study group has not looked at it yet still.

Jim: Did WG14 want it removed?

Fred: No, they definitely did not agree to that.

Jim: Then what did they approve?

Rajan: The implementation defined meaning of the macro and operations/operators.

Jim: Track the many changes to infinity and nan macros. Particularly still need to check prefixes for signaling nan macros. - Done.

Jim: Sent in the changes. New changes based on those will need to be reviewed again. Can close the item.

Last meeting action items (done unless specified otherwise, details below):

Rajan: Update CFP2341 to use paragraph form with semicolon separators.

Rajan: CFP2341: Look into seeing if we can say errno and the floating point environment can't be accessed (like with the "as if" rule).

Rajan: Update CFP2341 to say the identifiers for fenv.h and math.h should also be reserved as per stdlib.h in alternative 1.

Fred: Post the list of issues to be revisited for the next version on the CFP wiki.

Fred: Add an issue for a future version to look at the 5.2.4.2.2 terminology and ensure it is correct and consistent.

Fred: Put the NAN not raising exceptions footnote idea into the list of items for a future revision of C.

New action items:

Fred: Expand the individual items so they are self contained in C26.TXT.

Fred: Ensure that CFP2350 was sent to the C editor.

Jim: Respond to Vincent's CFP2368 message regarding the floating-point model.

Jim: Reply to CFP2371 saying it is in the Annex we proposed and WG14 voted in.

Upcoming WG14 meetings:

January 31-February 18 - Ongoing right now (one week meeting, one week off, one week meeting)

May 16-20 virtual

July 18-22 virtual

Action item resolutions:

Rajan: Update CFP2341 to use paragraph form with semicolon separators.

Rajan: CFP2341: Look into seeing if we can say errno and the floating point environment can't be accessed (like with the "as if" rule).

Rajan: Update CFP2341 to say the identifiers for fenv.h and math.h should also be reserved as per stdlib.h in alternative 1.

See CFP2346 and follow-ups.

Jim: The wording issues with parallelism and all was cleaned up well. But the various alternatives are really different things. There is no issue really so it is on the C++ side or Josephs side to bring up why they believe that locales or errno are an issue. For the floating point environment, the FENV_ACCESS pragma needs it's state to be ON before it can access the modes. If we say freestanding has it OFF or does not turn it on, it handles that aspect. The allowing math_errhandling to be zero is a worthwhile convenience that is independent. I believe it was an oversight on our part to not give the option for exceptions for the conversion functions.

Fred: Prefer Jim's method.

Rajan: Will go forward with Jim's method.

Jim: The footnotes seem to be implementation guidance. That is something the C standard does right?

Rajan: Yes.

Jim: Don't understand inclusive/exclusive and alternatives.
Rajan: That will go away with the new wording.

Fred: Post the list of issues to be revisited for the next version on the CFP wiki.
Fred: It is on the wiki: C26.TXT (https://wiki.edg.com/pub/CFP/WebHome/C26.TXT)
Damian: Which directory is it in?
Fred: As far as I know, there is only one directory in the wiki.
Jim: Is saying the document number good enough for things like the CFP2256 item.
Fred: I can write it out.
^Fred: Expand the individual items so they are self contained in C26.TXT.

Fred: Add an issue for a future version to look at the 5.2.4.2.2 terminology and ensure it is correct and consistent.
In the C26.txt file on the wiki.

Fred: Put the NAN not raising exceptions footnote idea into the list of items for a future revision of C.
In the C26.txt file on the wiki.

Other issues:
Terminology: floating constant vs floating-point constant (See CFP 2345,2349,2350)
Jim: Did you send CFP2350 to the editor?
Fred: I need to check. I believe I did.
Jim: Can you double check?
^Fred: Ensure that CFP2350 was sent to the C editor.
Damian: So we should use "floating constant" now in our documents?
Jim: Yes. It is a term in the C standard, not the general English term.

Subnormals and infinities (See CFP2368)
Jim: A C object representation that has a bit pattern that is a 754 subnormal need not be a C subnormal. An implementation can make all 754 subnormals as zero's.
Mike: Even for decimal?
Jim: No, since that has to follow IEEE.
David H: It's a little odd, but a reflection of reality. To change that would mean invalidating some implementations.
Jim: Right.
Jim: For the -infinity part, C does have "<" results in 1 if the relation is true. It doesn't explicitly say 1<2 just like it doesn't say 1 < infinity.
Fred: The C standard does say if -infinity is there the range is extended to all negative numbers. Ditto for positive.
Jim: I think that is consistent with it. The range type is a specific term used in a specific way. I am not sure he would agree that this says this. I don't think we need to say anything here. It is complete just as it is for real numbers.
^Jim: Respond to Vincent's CFP2368 message regarding the floating-point model.
David H: Does C allow an unsigned infinity?
Jim: It does.
David H: In the original 754, it was unordered.
Mike: Like NAN.
David H: Exactly.
Damian: How do you get one?!
David H: It was added. It wasn't IEEE.
Jim: Do implementations have unsigned infinity?
David: I've never heard of any.
Mike: You can ask Vincent if he knows any.
(Mike dropped off)

Others (new items added to the agenda):

See CFP2371.

Rajan: All we need to say is that it's been voted in and it'll be there when it is there.

^Jim: Reply to CFP2371 saying it is in the Annex we proposed and WG14 voted in.

Using the 'i' suffix for imaginary numbers:

Fred: A user, but not a standards member asked for the i suffix for an imaginary floating constant. I don't know if it is something we can get into C23.

Rajan: We've missed the cutoff.

Jim: Not critical or an IEEE-754 binding issue.

Fred: I'm in favor of adding it. Multiplying by 'I' is not always the cleanest way to do things.

Jim: Trying to think why we didn't consider this. Probably because we didn't want to change the language.

Fred: Imaginary is not required.