

WG14 N2949
Meeting notes

C Floating Point Study Group Teleconference

2022-03-16

8 AM PDT / 10 PM EDT / 3 PM UTC

Attendees: Rajan, Jim, Fred, David H., Ian

New agenda items (<https://wiki.edg.com/pub/CFP/WebHome/n2943.pdf>):

None.

Next Meeting(s):

March 30th, 2022, 3 PM UTC

ISO Zoom teleconference

Please notify the group if this time slot does not work.

WG14 meeting: See CFP2387

N2931: Accepted without issue.

C++ Liaison:

Nothing new.

C23 Integration:

Nothing new.

Carry-over action items:

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

Fred: Have to rewrite our current paper (remove the part already in C23) and submit it to WG21.

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

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.

New action items:

Rajan: Put it into the WG14 CFP report the outside request to make 7.12.13 and F<x>'s titles "Floating"->"Fused" that is editorial.

Fred: Post to CFP the WG14 schedule for C23: N2864. - Done.

Fred: Post the decisions on *HAS_SUBNORM by WG14 minutes to Jim.

Fred: Look at updating the C26.TXT file to follow what Mike is doing for 754.

David H: Get an example for the scaled reduction functions (perhaps by asking Jason or Jim or looking into the IEEE references).

David H: Get an example for the augmented arithmetic functions (perhaps by asking Jason or Jim or looking into the IEEE references).

Upcoming WG14 meetings:

May 16-20 virtual

July 18-22 virtual

Action item resolutions:

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

Fred: This item does not ring a bell.

Jim: The action item was to have a statement on what the issue was in the errata or the future issues list.

Fred: It was updated and posted on the wiki.

^Jim: I looked at that. Mike is doing a similar thing for 754. It may be helpful to look at how he is presenting that.

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

Fred: Not sure if I did it. Even if I did, the editor has never responded to any of my emails.

Jim: This was just to confirm you sent it.

Fred: I'll check. Otherwise, carry over.

Fred: I sent it on January 19th.

Jim: Respond to Vincent's CFP2368 message regarding the floating-point model. See CFP{2368-2386} interspersed.

Jim: Vincent said we have made it explicitly allowed that subnormals can be treated as zero by the implementation in defined cases. If there are subnormals in the model, they should be treated as non-zero. I said it is to allow existing implementations to continue doing what they are doing.

Jim: Reply to CFP2371 saying it is in the Annex we proposed and WG14 voted in. See CFP2371,CFP2373

Jim: I did respond on this.

Other issues

N2823 (freestanding) revision

Rajan: Still in progress. Should be done by next meeting.

fma subclause title [Cfp-interest 2388]

David H: We called it fused since there were implementations that had fused and unfused multiply-add's.

Jim: In C it is fused in the sense of only one rounding. The title says "Floating" though and Pavel wants it changed to "Fused".

Fred: The ARM hardware does double rounding.

Rajan: So they are already non-conforming?

Fred: Yes.

Jim: Is this our paper to do?

Fred: Can we make this an editorial change?

^Rajan: I can put it into the CFP report as an outside request that is editorial. I can ask WG14 what they want to do there.

TS 18661-4 update [Cfp-interest 2390]

Jim: Comments (labelled "[ISSUE]") throughout the document.

Jim: For the title, all that is left is the augmented arithmetic and reduction functions so it is very different from the original version of part 4.

Fred: The IEC 559 should be IEC 60559. (1989 version)

Jim: No, that was what it was named.

Fred: It was renamed.

Jim: It was the name on the website, but if you have something official showing an updated name, I can use that.

Jim: I want careful review on the introduction background both for the floating point and the C part. I was particularly hoping, David, that you would check the floating point part.

Jim: Should annex F conformance be required? The original part 4 did require it for binary or decimal or both.

Rajan: Is there anything remaining that has something that would not work on known floating point implementations (outside IEEE)?

Jim: The way this is written we have parts that do not need Annex F, and additional specification that does. For augmented arithmetic, the formats are the same as for 754 in that errors are report by floating point exceptions.

Rajan: The safest approach is to still require Annex F as implementations can do this without Annex F support if they want, and it won't break anything.

Jim: N2274 included tgmth macros for the augmented arithmetic. Should that be include in this document?

Rajan: I would suggest not having the tgmth functions but ask WG14 if they want it.

Jim: Should intermediate overflow/underflow be prohibited for non-IEC 60559 Implementations? We say "should".

David H: Is it useful to have these interfaces even if they are not going to do it any way other than what is fastest? We could have a pragma type construct to allow ignoring intermediate over/underflows with the default being not to ignore it.

Jim: FE_ENV_ACCESS OFF won't let you test for that so we already have it. If it overflows though it may not be appropriate for the final result. The pragma does not address that.

David H: The reduction functions were there to guarantee the operations were done a certain way even if inefficient. Maybe make it prohibited for all implementations but have a note about adding a method to allow it for all implementations without necessarily proposing it. Another proposal for that though. If you are doing this in a non-IEEE system presumably you are doing it this way due to the properties in the specification vs some other way.

Jim: 6.8 has the 60559 things like exceptions and NaNs and Infinities.

Jim: Is there enough prior art/intention to implement and use this feature? David, I remember you asked about this before. Any ideas?

David H: There was definite intention to use, and written up in some ARITH meetings. Someone did say they wanted to implement it in actual hardware, but whether or not it happened I need to ask again.

Jim: Section 7: Should these be there for non-60559? My sense is that these are there for reproducible results. And the proof of that depends on the subtleties of the format.

David H: I don't know how non-IEEE implementations would do this. I suspect they would relax a lot of places.

Fred: Without subnormals, I don't see how you could do it.

Jim: This is not whether you support Annex F or not, more if you support certain features. I think we're agreeing that we are requiring IEEE formats and more since subnormals have to work.

Jim: Examples of each (scaled reduction, and one for augmented arithmetic) would be useful.

Fred: How about a sum of two products. Needed for complex multiply and divide.

David H: More for larger arrays. Quotients of products of factorials. Important for physics. Mention this is a small part of a larger calculation and have a pair of factorials. The factorials would be computed in a scaled fashion. For augmented, we can reference papers since the examples would be too big.

Jim: For the scaled reduction functions, can you (David H) do that?

David H: Yeah, I can do that and then you can correct the C syntax errors.

Jim: For the augmented arithmetic, could they be tools for double-double?

David H: Yes, that was part of the intention. We can ask Jason if we need to come up with something.

Jim: Can we use them for sum of products?

David H: If you wanted to do the equivalent of dot product then rounding to single, that could be done. I thought we addressed this in one of the reference papers for 754 (obviously not in C syntax).

Jim: I didn't see anything in the ARITH papers that Jason or Jim wrote and could not see it. But I may have missed something. Can you (David H.) ask them about it?

^David H: Yes, I can.

Fred: For complex multiply and divide, you need the sum of two products. These functions seem the right way to do it.

David H: Yes, that would be a small compact example.

Using the 'i' suffix for imaginary numbers [Cfp-interest 2374,2375,2389]

Jim: I haven't seen a proposal for this.

Fred: It's an idea someone mentioned. If an implementation did not support imaginary types, what does it do for this suffix?

Jim: My sense was that it gives prettier syntax. $1.23i$ vs $1.23*i$.

Fred: You can't suffix a NaN or an Infinity.

Jim: No proposal so nothing for us to respond to here.

Fred: Does this remain binary or add in decimal?

Jim: 754 only did binary.

David H: Traditionally decimal doesn't need these. Mainly for scientific computing.

Others?

None

Rajan Bhakta