**WG14 N2812**
**Meeting notes**

## C Floating Point Study Group Teleconference

2021-08-17
8 AM PDT / 11 AM EDT / 3 PM UTC

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

**New agenda items**:
None.

**Carry-over action items:**
Fred: WG14 N2714 Add "a" before "NaN" in last bullet. - Done.
Jim: [CFP 1997]: Range error definitions of overflow and underflow.

**Last meeting action items (done unless specified otherwise, details below):**
All: Review CFP 2060.
Fred: Submit CFP 2062 to WG14.
Fred: Write up a proposal to remove the *_HAS_SUBNORM macros.
All: Look into the email history to find out why we chose to make float and _Float32 separate and distinct types.
Rajan: See what we can propose for freestanding and IEEE 754 and send it to the CFP mailing list.

**New action items:**
Jim: Update N2746 with CFP 2090.
Fred: Send CFP 2094 to WG14.
Rajan: Ensure the C/C++ study group presentation sees P1467r4.
Rajan: Draft words for making freestanding support for CFP for both options in CFP2085.
Jim: Send CFP2089 as an update to N2672 barring any issues from this group.
All: Look over CFP2096 and give feedback within 2 weeks.

**Next Meeting(s):**
Same time slot. Note: Back to original day of the week.
Wednesday, September 29, 2021, 3 PM UTC
ISO Zoom teleconference
Please notify the group if this time slot does not work.

**C++ liaison:**
CFP C23 changes summary page for the C++ liaison study group
Rajan: Sent out. See details in action item below.

**C23 integration**
Latest C2X draft: http://www.open-std.org/jtc1/sc22/wg14/www/docs/n2{596,573,478}.pdf also as link on CFP wiki
Part 2
Part 3
Part 4ab
Part 5abcd
IEC 60559:2020 support

**Carry over action items:**

Fred: WG14 N2714 Add "a" before "NaN" in last bullet.
See N2714.
Close item.

Jim: [CFP 1997]: Range error definitions of overflow and underflow.
See many CFP messages (Ex. CFP 2038-2090).
Re CFP 2090:
Jim: This would be a replacement of N2746.
Fred: An exact subnormal is an underflow.
Jim: Yes.
Fred: In 754 it may not raise the underflow exception.
Jim: Correct.
Fred: The definition of normal would need to change.
Mike: Is exact infinity defined somewhere?
Jim: The meaning is that the result is an infinity and is exact. An inexact infinity results from an overflow.
Mike: Need to check the uses of exact infinity and see if it is clear enough in the C standard.
^AI: Jim: Update N2746 with CFP 2090.

**Last meeting action items:**

All: Review CFP 2060.
Rajan: On the agenda for the C/C++ compat group (Meeting on Friday September 10th, 1pm New York time via Zoom).
See P2423R0 (https://wg21.link/p2423r0) C Floating Point Study Group Liaison Report

Fred: Submit CFP 2062 to WG14.
Done as N2790.

Fred: Write up a proposal to remove the *_HAS_SUBNORM macros.
See CFP 2094-2095.
Fred: Remove the *_HAS_SUBNORM macros and add in a new paragraph.
Fred: Since I wrote this, there were cases of flushing other than comparison operators!
Jim: The goal is to bring various subnormals into the standard and legal.
Mike: There was an old logarithmic thing that didn't have a zero.
Fred: A long time ago, C outlawed logarithmic floating point.
^AI: Fred: Send CFP 2094 to WG14.

All: Look into the email history to find out why we chose to make float and _Float32 separate and distinct types.
See CFP2074, CFP 2075.
CFP2075 seems to indicate there is no issue having float and _Float32 be different types.
Rajan: Depends on how P1467r4 was received by C++.
^AI: Rajan: Ensure the C/C++ study group presentation sees P1467r4.

Rajan: See what we can propose for freestanding and IEEE 754 and send it to the CFP mailing list.
See CFP2085/7/8.
Mike: Decimal separator (vs point/placeholder).
Jim: strtod not allowing exceptions, it was odd.
Rajan: The whitespace issue is still an issue.
Jim: Is there market demand for this?
Rajan: I don't see it for the large machine set, but have no insight into the small machine set.
Jim: Can you (Rajan) draft words for both of the options?
^AI: Rajan: Draft words for making freestanding support for CFP for both options in CFP2085.

Jim: Add detect tininess before and after rounding via macro to the meeting agenda.

Fred: I see no utility to having a macro for this.

Mike: I agree. Too esoteric.

Damian/David: Agree.

Damian: Doesn't the standard say the order?

Jim: Yes, it is pinned down for decimal, but not for binary. 754 didn't provide a mechanism for this either.

David: The only one who would care would be Fred. Don't see any other use for that information.

**Other issues:**

Typo in 5.2.4.2.2 (See CFP2083, CFP2089).

Fred: I don't think having all x with f1>0 is clear. Double-double.

Jim: But that is not a normalized floating point number.

Fred: Yes.

Jim: Can we attach this to the cleanup paper? Any issues should be brought up via email.

Fred: Sounds good.

Jim: If I get a positive response, I can add this on to the other paper.

^AI: Jim: Send CFP2089 as an update to N2672 barring any issues from this group.

Number classification and normal numbers (See CFP2091-3, CFP2096).

Fred: If the first digit is a zero, it is not normalized.

Fred: For CFP2096, double double can have values larger than the largest finite normal numbers.

Jim: Yes, this adds them to the normal numbers category.

David: We have normal numbers and subnormals, on the hand we have normalized and unnormalized. Subnormal means below normal. There is a lack of symmetry. The footnote can explain the difference between normalized and unnormalized.

Fred: There is also supernormal (double double has it). Do you know if DBL_MAX + DBL_MAX is a finite number instead of an infinity.

Jim: Can include implementation defined values that are not normal or subnormals.

Mike: Any finite number that is not a subnormal is a normal number.

Fred: That has the zero issue.

David: Can say "non-zero" finite number. Zero is clearly normal.

Jim: Zero is not normal. In 754 either. It is mutually exclusive classifications. Zero is a classification.

Mike: I want to get rid of the normalized part.

Jim: It's still in the subnormal definition.

Fred: It might make sense to do it Mike's way.

Mike: We do it this way in decimal. Since nothing is normalized in decimal, it never needs to be said.

Jim: So how do you define subnormal?

Fred: C already defines subnormal floating point numbers before this.

Jim: But not subnormal numbers (just floating point numbers).

David: In some ways the minimum value is implementation defined. Ex. 2 sub-norms that add up to the minimum normal number. Too many possibilities. An implementation could make everything normal numbers. If they are not claiming IEEE conformance, who cares?

Jim: The context here is we are giving the C floating point model. We don't want to throw it away.

Mike: We shouldn't define it in terms of normalizing. It mixes up two concepts that just happen to coincide for binary floating point numbers.

David: Is there something in the C model where we can definitely give a minimum normal value?

Jim: Normal is a full precision value in the type. The C model fits into the next clause, 5.3.4.2.3 where we talk about decimal, with an integer value and an exponent.

Mike: Since we don't need to use the term normalize for decimal subnormals, we shouldn't need it for binary.

Jim: For decimal, using the C model, it has a non-zero digit, followed by 6 more digits. The minimum normalized value would be 1 with 6 zeros, with an exponent (given in the C model).

Fred: DEC32_MIN definition has the word normalized in there.

Mike: Maybe that is wrong.

Jim: All of those refer to the C model. We need to see how this applies to the decimal model. This proposal is intended to address the double double format as well.

Jim: Vincent's issue for C17 having normalized representation being normalizable. For decimal, the extra semantics are representation level semantics, below the floating point model that talks about values.

Rajan: Not to sure where new proposals and bug fix timing fits into the C23 timing.

Fred: October 15th mailing deadline is the deadline for new proposals.

Jim: Can we put a time limit on this discussion so we can have it ready to go?

Mike: My comments are not 'we must do this', just that it can be improved.

^AI: All: Look over CFP2096 and give feedback within 2 weeks.

Fred: FLT_MIN should be FLOAT_TRUE_MIN / FLT_EPSILON for double double.

Regards,

Rajan Bhakta
z/OS XL C/C++ Compiler Technical Architect
ISO C Standards Representative for Canada
C Compiler Development
Contact: rbhakta@us.ibm.com, Rajan Bhakta/Houston/IBM