WG14 N1970
Meeting notes

C Floating Point Study Group Teleconference

2015-09-15
9 AM PDT / 12 PM EDT

Attendees: Rajan, Jim, David, Fred, Marius, Mike*, Ian*
*First hour only

New agenda items:
None.

Last meeting action items:
Ian: Talk to Michael Wong and Lowell regarding proposing this IEEE-754: 2008 binding to C++ as well - Done
Michael suggested talking with Lawrence Crawl (Numerics extension group in C++)
Question from them: Should _Float64 be the same as double in the typedef sense? We explicitly made it separate types. Has implications in name mangling and exception handling.
David: Part 5: Provide a mechanism (a new #pragma?) to allow implementations to possibly not propagate constant modes (rounding, exceptions) - Email sent out (May 20th, 2015) – Keep open - Drop the item

All: Review 7.6.1f.2 (in part 5, Aug 28) and report by email - Done (discuss below)
Jim: Change P 14#5 to say something like “for the listed exceptions” - Done
Jim: Write up change to disallow exceptions from appearing in more than one catch or delayed-catch list - Done
Jim: Try putting the input/results for the examples in a table - Done
Jim: Improve wording in last NOTE – change “delayed-catch” to “delayed-try and delayedcatch” – clarify that last statement applies to try-catch – change “should” to something like “might well be able to” - Done
Jim: Write up specification from email proposal about evaluation methods and math functions and include it in the draft - Done

New action items:
Ian: Talk to Lawrence Crawl regarding proposing this IEEE-754: 2008 binding to C++ as well - Still to do
David: Send out an email address to sign yourself up to the IEEE 754 mailing list to this group.
Ian: Update and check the items listed and flagged under Feature_List_Part_1.
Jim: Send Mike an email regarding what is needed regarding prior art/implementation for Part 1 features in other languages
Jim: p5: Give example of what the macro would do and what would happen without it.
Jim: p12: See if we can add a footnote regarding the implementation defined/unspecified/undefined behavior referring to Annex J and/or an example from one of the bullets ommitted.
Jim: p8: Make subnormal zero case be something that should keep the same sign
Jim: p15: line 31: ilogb -> ilogb and llogb
Jim: p15: line 32: FE_INVALID_LOGB -> FE_INVALID_ILOGB
Jim: p16: line 10: Remove 'and round result to narrower type'.
Jim: p9: FENV_ALLOW_CONTRACT_FMA: Send a note to convey this should not apply to any

implementation/system operations, and only to user code that is directly what is listed in lines 11-14.

Next Meeting:
October 13th, 2015, 12:00 EST, 9:00 PDT
Same teleconference number.
WG14 mailing deadline is September 28th.

Discussion:
Part 3: Target publication date: October 1st, 2015.

Part 4: No target publication date yet.

IEEE 754 group (new revision to the standard):
David still needs to schedule the first meeting.
Can subscribe yourself.
*David: Send out an email address to sign yourself up to the IEEE 754 mailing list to this group.

ARITH23 (conference next summer):
Marius: IBM, AMD, Intel presented new things in hardware last time. This TS could be discussed there.
Conference moving to every year instead of every other year. Does not have to be latest in research, but can be interesting things like state of the art hardware, etc.
The presentation could be the TS followed by the implementations by the vendors (Ex. IBM, Intel, GCC) and IEEE-754:2008 overview, etc.
David: Generally these panels want a diversity in points of view. Not interested in getting input for the 2018 version of the IEEE-754 standard, but would be for 2028.
Marius: Should I propose this? I will be there.
Jim: I can probably attend and present the TS.

Feature Lists:
*Ian: Update and check the items listed and flagged under Feature_List_Part_1.
Other languages may have these functions so we could use this under prior art.
Mike may know this from his test suite. *Jim to send Mike an email regarding what is needed regarding prior art/implementation for Part 1 features in other languages
Fred: GCC does not honor binary rounding mode when converting from DFP to binary floating point. It also has issues with rounding modes between decimal and binary floating point. These are partial implementations.

Part 5: Various emails, documents (cfp5-20150826.pdf)
p3/5: Examples would help. *Jim: p5: Give example of what the macro would do and what would happen without it.

Mike: 'BREAK' is not a good term since it is a keyword if lower-case.
Case matters so it seems fine the way it is.

Jim's September 9th email: Seems like a good approach
Perhaps have a footnote to reference one of the bullets we'll be pulling out and/or refer to Annex J?
*Jim: p12: See if we can add a footnote regarding the implementation defined/unspecified/undefined behavior referring to Annex J and/or an example from one of the bullets ommitted.

Zero subnormal sign issue (2015/09/14 email):
Make this a 'should' statement.
*Jim: p8: Make subnormal zero case be something that should keep the same sign

p15: Line 31/32: Should FE_INVALID_LOGB be FE_INVALID_ILOGB since the line above should refer to ilogb and llogb functions?
Suggestions: FE_INVALID_ILOGB, FE_INVALID_I_LOGB, FE_INVALID_INTEGRAL_LOGB
Change it to FE_INVALID_ILOGB.
*Jim: p15: line 31: ilogb -> ilogb and llogb
*Jim: p15: line 32: FE_INVALID_LOGB -> FE_INVALID_ILOGB

p16: Line 10: *Jim: p16: line 10: Remove 'and round result to narrower type'.

p9: Does ALLOW_CONTRACT_FMA apply to user code only or behind the scene code like complex mul and div?
Jim: The system code is a black box. This pragma should not affect that. Since the user has no clue what is in the black box, it should not matter.
Does C say anything about inline code being the same as something compiled in another CU?
This issue is not particular to this part of the TS. It is a wider question.
*Jim: p9: FENV_ALLOW_CONTRACT_FMA: Send a note to convey this should not apply to any implementation/system operations, and only to user code that is directly what is listed in lines 11-14.

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
