WG 14 N1935
April 2015 CFP Teleconference Minutes

2015/04/07, 9:00 AM PDT, 12:00 PM EDT:
Attendees: Rajan, Jim, Fred, Mike, Marius, Ian, David

New agenda items:
None.

Last meeting action items:
Jim: cfp5-diff-20150211-20150309.pdf: p4: Line 28: Check if 'indeterminate' is the right term. - Done.
Jim: cfp5-diff-20150211-20150309.pdf: p8: State what the default state is for #pragma STDC FENV_REPRODUCIBLE and what happens when it is OFF. - Done.
Jim: cfp5-diff-20150211-20150309.pdf: p9: Line 5: program code -> something else such as 'code sequence' or 'code segment' or 'Following are requirements for code in the whole program or part of a program'. - Done.
Jim: cfp5-diff-20150211-20150309.pdf: p9: Line 10: Note that we need to make sure functions with two or more arguments (since the order of evaluation of them is not fixed) is handled. - Still open.
Jim: cfp5-diff-20150211-20150309.pdf: p9: Line 32: The other fused multiply functions need to be listed here. - Done.
Jim: cfp5-diff-20150211-20150309.pdf: p9: Line 36: Remove "propagation, " - Done.
Jim: cfp5-diff-20150211-20150309.pdf: p9: Line 37: Look into tightening the underflow and inexact part. - Still open.
Jim: cfp5-diff-20150211-20150309.pdf: p10: Line 12: Change the statement to make sure that style 'a' applies to the entire statement. - Done.
Jim: cfp5-20150317.pdf: p10: Line 8: unspecified -> indeterminate - Done.
Jim: cfp5-20150317.pdf: p10: Line 16: unspecified -> indeterminate - Done.
Jim: cfp5-20150317.pdf: p10: Line 16: Remove duplicated text. - Done.
Jim: Part 5: Reproduciblity: Add in don't use alternate exception handling. - Not done. Not sure if this should be a restriction. Bring up for discussion.
Jim: Make the updates to part 5 and poll the group to see what to do for the C meeting. - Done (n1919 posted for the C meeting - Does not have alternate exception handling).

New action items:
Jim: cfp5-diff-20150211-20150309.pdf: p9: Line 10: Note that we need to make sure functions with two or more arguments (since the order of evaluation of them is not fixed) is handled.
Jim: cfp5-diff-20150211-20150309.pdf: p9: Line 37: Look into tightening the underflow and inexact part.
Jim: cfp5-20150330.pdf: p4: Line 20: FLT_EVAL_METHOD -> DEC_EVAL_METHOD
Jim: cfp5-20150330.pdf: p4: Line 32: actually -> actual
Jim: cfp5-20150330.pdf: p6: Line 23: Take into account rounding direction.
Jim: cfp5-20150330.pdf: p7: line 25 is misnumbered. It appears that one of the blank lines between 20 and 25 has a number.
Jim: cfp5-20150330.pdf: p7: Line 10 is a line number on a blank line
Jim: cfp5-20150330.pdf: p7: Last line: "NOTEIEC" appears to be missing a space. It has a tab.
Jim: cfp5-20150330.pdf: p7: Lines 31-32: Consider adding note about directed rounding
Jim: cfp5-20150330.pdf: p9: Lines 3 and 6: Consider clearer wording, maybe identifying IEEE/IEC section number
Jim: cfp5-20150330.pdf: p10: Line 43: NOEXCEPT: Like default, but no flag is set: wording improvement needed.

Jim: cfp5-20150330.pdf: p11: Line 3: OPTEXCEPT: Like default, but flag setting is optional: wording improvement needed.
Jim: cfp5-20150330.pdf: p11: Line 4: Reference IEEE for the meaning of 'tiny'. Also look into 'preliminary' and exact underflow.
Jim: cfp5-20150330.pdf: p11: Line 21: affect -> effect
Jim: cfp5-20150330.pdf: p11: Line 29: Remove "undefined behaviour" statement for every action that has a "shall" requirement already (BREAK, GOTO, DELAYED_GOTO).

Next Meeting:

May 19th, 2015, 12:00 PM EDT, 9:00 AM PDT
Same teleconference number.

Discussion:

WG14 meeting:
Rajan and Fred attending.

What should we do for alternate exception handling for the WG14 meeting?
We can post what we have, but not require comment or analysis from WG14.

Jim has been awarded the INCITS award for excellence for the work on these TS's. This should be brought up in WG14 to show the value of this work.

Part 1: Published.

Part 2: Published.
The flawed version was published due to changes from ISO before we had a chance to review their changes.
Jim is working with the Word draft that ISO used to get it fixed for something that can be published.
Issues with fonts, page breaks, line breaks, bullets, etc. all changing.
A similar thing happened with IEEE's spec.

Part 3: Ballot closed. Passed without comments.
Recommend publishing after editorial changes. See issues with Part 2.

Part 4: Ballot closed. Passed without comments.
Recommend publishing after editorial changes. See issues with Part 2.

Part 5:
Note: There will be duplicated notes at the end due to two separate note takers notes being merged. Both have been left to avoid missing anything.

cfp5-20150330.pdf:
*ToDo: Jim: cfp5-20150330.pdf: p4: Line 20: FLT_EVAL_METHOD -> DEC_EVAL_METHOD
*ToDo: Jim: cfp5-20150330.pdf: p4: Line 32: actually -> actual
p6: Should we change what is listed to put in the words to say "includes the following" or leave it as is?
Leave it as is.
p6: Line 22: Not true due to rounding directions. Needs to be changed. Can get rounding in the opposite direction.
Directed rounding users probably don't want any optimization so they would not allow this

optimization.
These optimizations are valid for real variables. Only invalid for floating point due to limits.
The rational is that they are equivalent in real arithmetic and not in IEEE representations.
This could mean we allow other properties of real numbers as well. Do we want that slippery slope?
Associative and Distributive seem to be different properies. The recipricol multiply is done by our compiler (ex. if it is a compile time constant and a power of 2, it is always done). This is different from line 7 but similar.
It is the same with the associative law.
Prefer two separate pragma's but fine with this the way it is.
Will impact existing compilers to determine what overrides what.
*ToDo: Jim: cfp5-20150330.pdf: p6: Line 23: Take into account rounding direction.

Page 6 lines 23-25 distributed law over subtraction:
Jim: You can do this in non-directed rounding modes.
David: Not true in any rounding mode, but this pragma permits it.
Minuses introduce other issues.
This pragma allows doing it.
Assume anyone using it either doesn't use directed rounding or doesn't mind the change.
Lines 5-7 are similar.
Page 7 line 25 is misnumbered. It appears that one of the blank lines between 20 and 25 has a number. Jim to fix.
Page 7 line 10 is a line number on a blank line. Jim to fix.
Page 7 last line "NOTEIEC" appears to be missing a space. It has a tab. Jim to fix???
Jim: Change bars sometimes appear to apply to a line above or below the changes.
Page 7 lines 31-32:
Jim: The last one "x-y*z" may not be valid with directed rounding.
David: OK? One who cares can use fma function.
Fred: x-y*z should be z-x*y.
Jim: Not saying these expressions are equivalent, so OK as is.
Jim to consider adding note about directed rounding.
David: Maybe list all expressions on line 26?
Page 8 Joseph asked about assignments and function call results, so Jim expanded example.
Ian: Also function parameters [needing conversion]? Jim: Yes.
Also includes clarification Fred requested.
Page 9 line 3: Only equivalent if implementation supports pragma.
(Skip section 9 Alternate exception handling for now.)
Page 13 lines 29-33:
Jim disagrees that reproducibility excludes all alternate exception handling, so rewrote this to be more specific.
Page 14 lines 14-15 character sequence conversions:
The conversion functions are IEEE/IEC operations, so need mentioning.
Page 9 section 9 Alternate exception handling:
Jim added text for context.
Wrote draft of pragmas equivalent to try/catch suggestion.
Is action the right word? Ian and David: Yes.
Page 9 lines 3 and 6:
David: Why mention "to narrower type"?
Jim: Add rounding to narrower type identifies which IEEE/IEC functions this applies to.
Jim to consider clearer wording, maybe identifying IEEE/IEC section number.
Other places have similar wording.

Page 10 lines 39... Action:
Page 10 lines 42-43:
David: NOEXCEPT is like default except flag must not be set (could be expensive).
Jim to improve wording.
Page 11 lines 1-3: Jim to improve wording.
Page 11 line 4 abrupt underflow: "tiny" means the same as in IEEE/IEC. Jim to reword or refer to IEEE definition.
Page 11 line 5: Fred: IEEE does not use "preliminary". Jim: Two roundings. Jim to improve wording.
David: IEEE defines numeric result, so don't need to do that here.
Jim to improve (simplify?) wording.
Page 11 line 21: "affect" => "effect".
Page 11 lines 28-29: Fred: Compiler can detect error, not make this undefined.
Rajan: Needs rewording that action shall not be used outside a compound statement.
Jim: Do any std macros have constraints?
Rajan: Upcoming defect may resolve this.
Rajan: Don't need last sentence. Jim to remove it.
Page 11 lines 32-33: Jim to remove "Use of the pragma with this action outside any compound statement results in undefined behavior."
Page 11 lines 40-41: Ditto.
Page 11 lines 43-47: Jim: Implementation?
David: Just generate code and test flag at end?
What if the only exception is an exact underflow which doesn't set a flag? Then it couldn't be detected.
Jim: That small if is a big if.
Ian: Does programmer really care about an exact underflow?
David: Sometimes you don't want a subnormal.
Jim: Issue between underflow, overlfow and exact when using default?
---
p10: Line 39: NOEXCEPT: Like default, but no flag is set.
This means that if hardware sets a flag, it would have to be unset.
*ToDo: Jim: cfp5-20150330.pdf: p10: Line 43: NOEXCEPT: Like default, but no flag is set: wording improvement needed.
p11:
*ToDo: Jim: cfp5-20150330.pdf: p11: Line 3: OPTEXCEPT: Like default, but flag setting is optional: wording improvement needed.
Abrupt: What does 'tiny' mean? IEEE defines it. Should we reference IEEE here?
Floating point standard doesn't use the word preliminary. Unrounded may be used. But that gets into before or after rounding.
We could remove the parenthetical and just refer to IEEE.
*ToDo: Jim: cfp5-20150330.pdf: p11: Line 4: Reference IEEE for the meaning of 'tiny'. Also look into 'preliminary' and exact underflow.
*ToDo: Jim: cfp5-20150330.pdf: p11: Line 21: affect -> effect
Break: Take out line 29 as the "shall" in the statement before already requires a diagnostic or failure to translate.
*ToDo: Jim: cfp5-20150330.pdf: p11: Line 29: Remove "undefined behaviour" statement for every action that has a "shall" requirement already (BREAK, GOTO, DELAYED_GOTO).
Delayed Goto: Default exception handling may need work.
Only applies to the exception list.
What if the only exception is an exact underflow (so can't be detected by checking the flag)?
What's wrong with an exact underflow in the first place? People don't want the subnormal?

Looks like we won't have anything for Part 9 that we want to bring forward to WG14 due to the large flux we're still going through working through the details.

Regards,

Rajan Bhakta
z/OS XL C/C++ Compiler Technical Architect
ISO C Standards Representative for Canada
C Compiler Development