WG 14 N1953
July 2015 CFP Teleconference Minutes

Attendees: Rajan, Jim, Blaine, David, Ian

New agenda items:
Discuss feature list for parts 1 and 2.

Last meeting 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. - Listed as unspecified behaviour in IEEE. In our spec we say in Annex J the program has to deal with it. Jim to send an email regarding this. - Done

Jim: cfp5-diff-20150211-20150309.pdf: p9: Line 37: Look into tightening the underflow and inexact part. - Still open. - IEEE says the program can't depend on inexact or underflow. David: May be due to before or after rounding. David: With underflow, flags and traps are not the same due to exact underflow. - Done (in draft)

Ian: Talk to Michael Wong and Lowell regarding proposing this IEEE-754: 2008 binding to C++ as well - Still open

Rajan, Jim: Talk to David Keaton regarding our intent for putting the essentials of parts 1 and 2 (not necessarily exact match) of this TS into the next C Standard. Need to look at reflector message 13739. - See parts 1 and 2 feature list discussion

Jim: Part 5: Add in examples of ASAP and Delayed Goto pragma's in the same block and in different orders. - An example with two delayed Goto's and text re ASAP was added. Look at try/catch to see if it affects this. - Keep open

Jim: Part 5: Add in a new issue - Is it worth adding expression evaluation methods that widen the library functions as well as the operators (that is already there)? - Email sent to discuss this on June 12th - Keep open

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

Blaine: Send out a DR process document so we can use that for Parts 1-4. - Keep open

Rajan: Create an outline of features in parts 1 and 2. - Done

Jim: Update the wiki to put in links to the test suites (from Mike) and possibly compiler manuals that address these parts. - Done

Fred: Find out the expiry date for an ISO document 60559. Expires July 2016. See email from Fred. - Done

Jim: Page 1: Fill in dates for parts 3 and 4. - Done

Jim: Page 3 and 4: Make the changes as proposed in the email on 2015/06/10 by Jim - Done

All: Do the FENV_ALLOW_* pragmas apply to complex types? During document review. - Done

Jim: Add a conformance macro for alternate exception handling. - Done

Jim: Word the goto description better to disallow jumping to a label for an exception that did not happen. - Done

Blaine: Send an email to continue the 'goto' discussion and issues related to it. May go away with try/catch. - Done

New action items:

Jim: Part 5: Page 2: alternate exception handling attribute -> alternate exception handling attribute*s*

Jim: Part 5: Add in words to make the FENV_ALLOW_* pragma's apply only to the floating types (which includes complex)

Jim: Part 5: Write up the paired immediate and delayed try/catch's
Jim: Contact Marius to see what Intel has done to fill in the Parts 1 and 2 feature list documents
Rajan: Contact Joseph to see what GCC has done to fill in the Parts 1 and 2 feature list documents

Next Meeting:
August 25th, 2015, 12:00 EST, 9:00 PDT
Same teleconference number.

Discussion:
Part 1: Published.

Part 2: 2nd edition has been published.

Part 3: Should go back to ISO today as publishable.

Part 4: Should go back to ISO today as publishable.

Part 5: Various emails, documents (July 5th draft - cfp5-20150705.pdf)
Page 2:
*ToDo: Jim: alternate exception handling attribute -> alternate exception handling attribute*s*
Page 3:
Blaine: Any precedence for this? Thinking of precompiled headers.
Jim: FLT_ROUNDS is like this.
Not the same since that is runtime.
This is compile time.
Can be done by having the pragma set the macro.
Precompiled headers should follow the standard so it should not be an issue.
Blaine: Pragmatically precompiled headers is often done regardless of dependencies.
Consensus: Keep as is.
Page 4:
Similar to library functions, and other identifiers being hidden by macros.
Should work for Generic selection as well as that is done after preprocessing.
Page 11:
Line 11: Handles argument ordering and other issues like this.
Page 12:
Line 10: Email sent on June 30th.
Blaine: The more complicated wording says if the underlying library supports it then you can count on it.
Rajan: Having the first proposal as a footnote makes the text incorrect (both proposals give different sets of reproducible programs). Theoretically the first way is correct, while the second is easier to read and likely not to have an issue in practice.
Consensus is to leave it as is.

Should ALLOW pragma's apply to the Complex types?
Ian: Should the CONTRACT_FMA one disallow the internal implementation of normal complex multiply?
Jim: There is no reproducibility for complex so it should be OK.
Ian: It could apply to integers as well?
Currently it seems to apply to complex.
*ToDo: Jim: Add in words to make the FENV_ALLOW_* pragma's apply only to the floating types (which includes complex)

Exception handling try catch in a pragma (try-catch-20150703.pdf):

For all the FENV_EXCEPT pragmas, put the TRY/CATCH/etc. words before the FE_* exceptions list.

Blaine: A simplification that doesn't lose anything

Catch can be turned into delayed catch for everything except underflow (though it would not be an optimal implementation).

Rajan: Add in a delayed_try to match with delayed_catch, and try with catch.

Better to have explicit nesting than implicit nesting.

Note that if you have a nested case (Ex. delayed_try overflow, try invalid [or swap the order], overflow happens, invalid happens. In this case both the invalid and overflow catch's handlers will run). Note the current way with nested try's gives the same issue, but a single try will not.

Can also have a list of both delayed and immediate exceptions in the 'try' list. Ex. EXCEPT TRY FE_OVERFLOW FE_UNDERFLOW DELAYED INVALID

Blaine: Conceptually simpler having the paired immediate and delayed try/catch's

*ToDo: Jim: Write up the paired immediate and delayed try/catch's.

Incorporating the TS into the C standard:

*ToDo: Jim: Contact Marius to see what Intel has done to fill in the Parts 1 and 2 feature list documents.

*ToDo: Rajan: Contact Joseph to see what GCC has done to fill in the Parts 1 and 2 feature list documents.

Regards,

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