WG 14 N1827

WG14 CFP meeting minutes for the meeting of 2014/03/18

2014/03/18, 12:00 EST:
Attendees: Jim, Rajan, David, Mike, Marius

New agenda items:
Timing for people available during the WG14 C meeting and means of contact.
Rajan will attend in person.
Jim will be on the phone when he can. Not available on April 11th.
The meeting time will be 12:00am - 3am, 4:30am - 7:30am Pacific time, 3-6am, 7:30-10:30am Eastern time
*ToDo: Everyone: Send Rajan available times for April 7-11th and contact information

Old action items:
Jim: Backup the documents in Word format. - Most current version has been put up. Keep this item open.
Current files: http://wiki.edg.com/twiki/pub/CFP/WebHome/cfp1.docx
Current files: http://wiki.edg.com/twiki/pub/CFP/WebHome/cfp2.docx
Current files: http://wiki.edg.com/twiki/pub/CFP/WebHome/cfp3.docx
Current files: http://wiki.edg.com/twiki/pub/CFP/WebHome/cfp4.docx
Note: Should also keep versions that are equivalent to PDF's.
- Standard practice now. Remove the item.

Jim: Part 3: Add explanation of type classification somewhere in the TS or changes to the standard document. - Done
Jim: Page 22: See about breaking the paragraph up to make it clearer (maybe bullets). - Done
Jim: Page 44: Remove dN and dNx since there are no complex decimal types. - Done
Jim: Part 4: Page 6: Talk to Fred to see if he is OK with the errors. - Still under discussion
Jim: Page 15: Move the cr prefix new text to a footnote. - Done
Jim: Page 23: Typos (changes 2 and 3): scaled_prod -> scaled_prodsum, scaled_prod -> scaled_proddiff - Done
Rajan: Ask WG14 why pole errors are specified the way they are in the C meeting. - Not done

Next Meeting:
April 15th, 2014, 12:00 EST, 9:00 PDT
Same teleconference number.

New action items:
Everyone: Send Rajan available times for April 7-11th and contact information.
David: I will take a pass at making the first draft of this part (without regard to format/template). I need someone familiar with C to give me the list of FP pragmas and their scope and effect.
Jim: Send David the C floating point pragma scope/effects.

Discussion:
Part 1: Out for ISO DTS ballot. Ballot ends March 5th, 2014.
No comments so far.
If comments come in before the meeting, we will have an email exchange to prepare recommended responses to the comments.

Part 2: Out for first ISO ballot (PDTS). Ballot ends March 10th, 2014.
No comments so far.
If comments come in before the meeting, we will have an email exchange to prepare recommended responses to the comments.

Part 3: (http://wiki.edg.com/twiki/pub/CFP/WebHome/n1796.pdf)
Ask for a WG14 review and subsequently a PDTS ballot.
If a version of the C standard that can be updated (ex. in LaTeX) can be made available, we can create a consolidated document.

Part 4: (http://wiki.edg.com/twiki/pub/CFP/WebHome/n1797.pdf)
Ask for a WG14 review and subsequently a PDTS ballot.
Not as many dependencies on previous parts as Part 3.
March 17th email exchange: "Too large" and "Too small" terms.
Can we change this to "Greater than" or "large positive"?
Differs from existing C11 practice and wording.
Jim: We have two choices if we want to change:
1) Change every instance in C11 to "Greater than"
2) Change Part 4 instances and live with the difference in wording from C11
No consensus to make any change.
Rajan: If this is a problem, it can be brought up as a defect report against C11 and we could follow whatever the response is.

Part 5: (Attributes email from Jim sent 2014/03/18)
Attributes are like static modes that affect a block of source code.
The FP pragmas are an example of that in C.
Affect a block or everything in the file.
We can use the FP pragma model for these attributes or any other form.
David: We can assign attributes to operations, though it will probably not be possible in C.
Mike: In Java, a context was added to each operation, but it is not obvious how this would be done for C.
WG14 did not want to go down the generalized attribute format as C++ did. They preferred keywords and pragmas.
The constant rounding mode pragmas do take parameters, but do not specify operation scope (ex. in this block, all multiplications will follow this pragma specification)
Note: OpenMP pragmas do follow the restricted scope pragmas so it should be easy for most implementers.
We have already done the constant rounding mode attributes through pragmas.
Exception Handling:
Detection of subexceptions:
Ex. Invalid due to SNaN, or due to 0 * Inf, or due to ...
Marius: Subexceptions can be described as priority of exceptions in some documents. Is this a better way to describe it?
David: Priority is a different issue. Ex. SIMD has two different exceptions in different data streams. Priorities determine which one is percolated.
Marius: What about SNaN / 0 and both cases have different exception handling?
David: We need to call it out. Not a part of the general Invalid exception. Another case could be underflow/overflow with inexact. We say somewhere that exponent has priority over inexact.
Jim: I think we also specify the SNaN / 0 case.
David: Ex. In some group of code you want 0 / 0 to result in Pi or 1 or whatever, but leave the other exceptions to the default handling. Another example is -ve sqrt since you may know it is only due to rounding so it can be set to zero, and other exceptions are valid.

Resuming alternate exception handling:
  Looks complete.
Exception timing:
  Delayed: Default exception handling for within the block, then special at the end of the block
  Transfer:
    Function call?
      Scope issues for parameters?
    Can't throw directly in C.
    Can do the delayed transfer to a function with what we have already through clearing the flags, run the block, test the flags and call a function based on the test results.
    Sub-exceptions would need to be worked out.
    How well does what we already have in C provide this? If it is not good enough, we will have to come up with something like a pragma to handle it.
Preferred width:
  Like evaluation method, but on blocks:
    Need to add something (ex. pragma) that specifies the evaluation method for the scope of the pragma
  Note: Some IEEE operations are C functions. The evaluation methods in C apply to operations, not functions. Ex. sqrt.
    With the type generics and _t types you should be able to get it for even C functions.
Value changing optimization:
  David: I can see the scope varying from small areas to large areas with some exceptions
  Jim: The FP_CONTRACT partially does some of these.
  Do we need more discrimination over what the FENV_ACCESS pragma gives?
  Ex. An accuracy pragma?
  You can have:
    1) a control/knob for everything you want to do
    2) a control/knob for group of things you want to do
    3) one control: Reproducible mode or non-reproducible mode
  David: Asking for standardization is against the conditionally normative approach
Reproducible results:
  David: A high level constraint on optimization.
    Would non-reproducible constructs be a requirement or recommended?
    Jim: If we can define it precisely, it could be something we can require diagnostics for.
  Marius: What is the domain of reproducible? Single thread, particular hardware, etc.?
    David: It should be for any conforming implementation? It would require specifying the degree of reductions (ex. 2 threads).
*ToDo: David: I will take a pass at making the first draft of this part (without regard to format/template). I need someone familiar with C to give me the list of FP pragmas and their scope and effect.
*ToDo: Jim to send David the C floating point pragma scope/effects.
Should apply to both BFP and DFP
How do the new types fit into this?

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