ISO/IEC JTC1/SC22
Languages
Secretariat: CANADA (SCC)

ISO/IEC JTC1/SC22
N 1111

FEBRUARY 1992

TITLE: USA Member Body Contribution explaining the USA "No" vote on CD 10967

SOURCE: Secretariat ISO/IEC JTC1/SC22

WORK ITEM: JTC1.22.28

STATUS: New

CROSS REFERENCE: CD 10967, N1010

DOCUMENT TYPE: MB Contribution

ACTION: For information to SC22 Member Bodies.
For action by WG11.

Address reply to: ISO/IEC JTC1/SC22 Secretariat
J.L. Côté
Treasury Board Secretariat
140 O'Connor St., 10th Floor, Ottawa, Ontario, Canada, K1A 0R5
Tel.: (613)957-2496 Telex: 053-3336 Fax: (613)996-2690

DOCUMENT:	
TITLE:	USA Contribution explaining the USA NO vote on JTC1/SC22 N935
SOURCE:	USA
PROJECT:	22.28
STATUS:	National Body Contribution
REQUESTED ACTION:	for review at the SC22/WG11 Meeting, April 1992 at Fort Meade, Maryland.

The following information is contributed to clarify the reasons for the U.S. NO vote on JTC1/SC22 N935 that proposed promotion of ISO cd10967 to draft international standard.

X3 initiated a public review of ISO cd10967, the LCAS. Comments received by X3 and forwarded to X3T2 in response to the U.S. review raised a number of concerns in the areas of notification, rounding, and compatibility with IEC 559 (IEEE 754). In light of this, the U.S. voted NO, but would vote YES if the following changes were accepted.

1. Clause 5, Notification, must be completely rewritten. Notification may include as many as five alternatives. A summary of the current proposal is as follows:

   (a) Use status flags corresponding to the four LCAS exceptional values: integer_overflow, floating_overflow, underflow and undefined (see items 5 and 6 below). IEEE 754 requires status flags which can be mapped onto the last three LCAS exceptional values. Any implementation which supports interrupts can implement such flags in software.

   The flags must be cleared at the start of a program, and get set by the implementation on occurrence of a violation. The implementation must provide callable routines to access the status flags. It must also provide routines to save and restore the entire set of flags. The names, arguments and output for all of these routines will be specified.

   This alternative allows a program to test the flags at strategic points, and then, if indicated, to invoke a handler for the violation or take other corrective action.

   (b) Prompt delivery of a message, followed by continuation of execution.

   (c) Prompt delivery of a message, followed by termination of execution.

   (d) Prompt alteration of control flow, such that execution shall continue only as a result of explicit action specified in the program.

   (e) Prompt execution of a program defined handler, followed by continuation of execution.

   Delayed notification is permitted. Continuation values for alternatives (a), (b), (d), and (e) are implementation dependent for the overflow and undefined flags. For underflow, the continuation value is rndF(exact_result) when denorm is true and is zero when denorm is false.

2. Clause 4.2.4, Range Checking.

   The range checking logic must be modified to reflect the changes in the notification procedure.

3. Clause 4.2.3, Rounding. Add the condition that

   ```
   rndF(-x) = -rndF(x)
   ```

   Remove mention of other rounding rules.

   Require an implementation to supply a parameter giving the error bound of rndF as a multiple of ulps in the returned result.

   List the additional identities satisfied by addF, subF, mulF, and divF.

   ```
   addF(-x,-y) = -addF(x,y)
   subF(-x,-y) = -subF(x,y)
   mulF(-x,y) = mulF(x,-y) = -mulF(x,y)
   divF(-x,y) = divF(x,-y) = -divF(x,y)
   ```

   in Annex A.4.2.9.

4. Clause 4.3, Conversions.

   A rnd-nearest rounding function shall satisfy

   ```
   |rnd-nearest(x) - x| <= (r^(e(x) - p))/2
   ```

   Then require that conversions from integer to floating point and from one floating point type to another use a rnd-nearest rounding function.

   Note that this definition of rnd-nearest allows either direction of rounding for the "half-way" case.

5. Clause 4, The arithmetic types.

   Split the exceptional value overflow into two values integer_overflow and floating_overflow.

6. Clause 4, The arithmetic types.

   Remove the exceptional value zero_divisor, and replace it by undefined in Clauses 4.1.1, 4.1.3, 4.2.2, and 4.2.6. This change is needed for compatibility with the treatment of the division by zero exception of IEEE 754.

7. Clause 4.2.6, Axioms.

   Change the signF(x) axioms to signF(x) = -1, 0, and +1 for x negative, zero, and strictly positive, respectively.

8. Clauses 4.2.2 and 4.2.6 and Annex 4.2.2

   Remove sqrtF from the LCAS and include it instead in the Language Compatible Mathematical Procedure Standard, also under development as an international standard.

9. Several editorial corrections requested by the comments will be forwarded to the editors.

The proposed changes to ISO cd10967 currently being circulated by WG11 include most of the changes requested above. The single exception is that the change described in 1(a) above does not require checking of the status flags for successful completion of the program.