This document contains responses to the comments received on CD10967-2 (SC22/N2132) in connection with the ballot for progression to CD. Comments were submitted by four National Bodies, Japan, Sweden, the United Kingdom, and the United States. Also comments were submitted by Frank Farance and Fred Tydeman, as individuals.

The responses appear below as follows:

Response to Japan's Comments on ISO/IEC 10967-2

These comments state that the National Body of Japan disapproves IS0/IEC CD 10967-2 for reasons given in the comments.

Summaries of Japan's comments, together with responses, follow.

****

1. Error Limits:

It appears that LIA-2 requires that an implementation provide functions with the smallest possible error limits. This would contradict Japan's understanding that LIA-2 was intended to allow an implementation to choose how to make the tradeoff between accuracy and performance.

> Japan's understanding is correct. The functions as currently specified provide high, but not maximum, accuracy. It is intended that an implementation provide a library of reasonably high accuracy, and also provide one of lower accuracy, with error bounds three or four times greater than those for high accuracy. Maximum accuracy is half an ulp for most functions; LIA-2 requires this only for SQRT.

****

2. Definition of Conformity:

The target of the standard appears to be programming languages. Can non-language entities conform?

> Yes. Such conformance is implicit in Clause 2.

The term "binding standard" is defined in Clause 2.2; however, the existence of binding standards is not required. Under what circumstances is a binding standard necessary?

> A binding statement is always necessary. However, it is not really suitable for LIA-2. Rather it is better standardized by the language, which may include only a subset of the operations specified in LIA-2.

In Clause 7(b, d, f) an implementation is required to document notations for expressing some entities. Rather than being defined by each implementation, they should be defined on a per language basis.

> This would imply that the language committees have the responsibility of providing such binding standards.

****

3. In clause 4.1, in the definition of s[i], "S" should be replaced by "s" in three places.

> DONE.

In the definition of rounding functions, an integer type is written as S, but the standard requires I for this purpose.

The symbol S is used to refer to any datatype (including integer) in this clause. That is the floor, ceiling, nearest, and truncate functions are defined for any datatype.

At the beginning of the clause, it is stated that Z denotes the set of mathematical integers. The use of I for an integer datatype is introduced later in this clause. Both of these notations are in reasonably common use.

****

Clause 5.2:

Items (b) and (e) do not use the word "shall" even though they are requirements.

The word "shall" now appears.

****

Clause 6.1:

This clause defines the continuation value for "undefined" to be a quiet **NaN**. The continuation value is given as zero in clause 9.5, and as +infinity in clause 10.1.

The phrase "unless explicitly specified otherwise" has been added to the definition in clause 6.1.

****

Clause 7: "must" should be changed to "shall" in the paragraph just after the note.

DONE.

****

Clause 8.1 (SQRT):

The condition of the exception (if $x < 0$) should be given.

DONE.

Error limits in general: Two units, ulps and rnd_error are used.

That is OK: rnd_error is defined in terms of ulps.

****

Clause 9.3 (POWER_FF):

In the table of extensions $0^{-\infty}$ and 0^{minus} are defined to be +infinity. We suspect they should be "undefined." Also $1^{-\infty}$ and $1^{+\infty}$ are defined to be "undefined." We suspect that they should be 1.

These extensions are influenced by various references dealing with the proper way to handle the extensions. Since they are essentially defined by a limiting process involving a path in two dimensions, many depend on the path selected. LIA-2 uses "undefined" to indicate such ambiguity.

In Note 1, it is stated that values corresponding to dashes in the table are given in the axioms of exceptions component. This is not true. Only special values are given in the axioms component. Usual values are specified in the definition component.

> The note has been suitably reworded.

****

Clause 9.4, POWER_FI

0^{minus} is defined to yield "undefined." We suspect it should yield "**pole**."

> This was not done for consistency with $POWER\_FF$.

The condition $y > 1$ for underflow is not quite correct. Underflow might occur for $y < 1$. This case should be added.

> In $POWER\_FI(x, y)$, y is an integer. The operation is undefined for $y < 0$.
>
> There is no underflow for $y = 0$, the only remaining value of y less than 1. Hence, there is no underflow for $y < 1$.

"|POWER" should be "| POWER"

> "|POWER" does not occur, but "IPOWER" does.
>
> At present the first line of the extensions is unclear. It has been revised, so that the meaning is clear.

****

Clause 9.5, POWER_II:

The axioms section is unnecessary.

> Yes, but it does no harm and fits well with the first exceptiom.

$power(x, y)$ is undefined if y is negative. Is this true even in the case $x = 1$, $y = -1$?

> Yes. We could have added specifications of $y < 0$, but considered them of insufficient utility to justify the additional burdens on implementors for implementation and verification of correctness.

****

Clause 10.3 LOG_FF:

Logarithm is an increasing function if $b > 1$. But it is a decreasing function if $b < 1$. This fact is not taken into account in some definitions (first line of axioms, fourth line of exceptions, and fifth line of extensions).

> This has been fixed.

The second and fifth lines of the exceptions duplicate information.

> The duplication has been removed.

****

Clause 11.3, COS_F:

An axiom $COS\_F(0) = 1$ should be given.

NOT DONE – this axiom is a special case of $cos(x) = 1$ for “small” x.

****

Clause 11.5 TAN_F

Remove “)” from ”$< fmin_N$)”

Instead, $fmin_N$) has been replaced by r^{-p}.

The underflow condition is not consistent with the underflow condition for SIN_F.

The underflow condition has been corrected.

****

Clause 11.6, TAN_FF:

The operators “==” and “mod” are not used in other sections.

They have been removed from this section.

In the fifth exception “$*(x)$” should be “$*x)$”

DONE.

****

Clause 11.7, COT_F:

The case $cot(x) > fmax$ and $|x| < fmin_N$ is missing.

This case has been added to both COT_F and COT_FF, where it is also missing.

****

Clause 12.4, ARCCOS_FF:

The fourth axiom is not necessary; it can be derived from the last axiom.

It does no harm, and completes the special case axioms. Hence it has not been removed.

****

Clause 12.8, ARCTAN2_FFF:

Only the case $y/x > 0$ is considered for underflow. Underflow may occur for $y/x < 0$.

This has been corrected.

****

Clauses 12.10, ARCCOT_FF:

The specification for underflow is missing.

The underflow specification is now included.

****

Clauses 12.13, and 12.14, ARCSEC_F and ARCSEC_FF:

The specifications for underflow are missing.

The corresponding functions are never zero; hence underflow is impossible.

****

Clause 12.14, ARCSEC_FF:

The axiom $ARCSEC(-1, u) = u/2$ is not necessary; it can be derived from the last axiom.

It has been removed.

****

Clause 12.17, Inverse Trigonometric Operations in Degrees:

ARCCOT2_360 should be given.

This has been done.

Note that the specifications for the degree operations have been expanded, and that their positions in the list of operations have changed.

****

Clause 13.1, SINHf:

The axiom $SINH(0) = 0$ is unnecessary; it can be derived from $SINH(-x) = -SINH(x)$.

It has been removed.

****

Clause 14.1, ARCSINHf:

The axiom $ARCSINH(0) = 0$ should be replaced by $ARCSINH(-x) = -ARCSINH(x)$.

DONE.

****

Clause 14.2, ARCCOSHf:

The lower bound of $ARCCOSH$ should be 0, not 1.

This error has been corrected.

The condition for "undefined" should be $x < 1$, not $|x| < 1$.

This error has been corrected.

****

Clauses 16.1.1 and 16.1.2, TRUNCATE_INf and NEAREST_INf:

In the extensions components, these functions are used with three arguments. Each takes only one argument.

The last two arguments have been removed in each of these operations.

****

Clauses 18.3.1 and 18.3.2, SUMHIf and SUNLOf:

The conditions for exceptions are not fully specified.

These operations have been extensively revised.

****

Clause 20.4, GCDi

The set membership operator should be used in "ninZ".

This has been corrected.

Response to Sweden's Comments on ISO/IEC 10967-2

1 Introduction

Sweden classifies the issues in the first section of its comments as follows:

a) Very major: definite cause for a "No" vote unless resolved in a satisfactory way.

b) Major: cause for "No" vote, future vote reserved, but likely to be "No" if not handled properly.

c) Minor: not cause for "No" vote, but should be handled properly and can still be argued strongly for.

These are followed by a section containing "clause-by-clause" comments. Finally, there are two additional sections presenting Sweden's proposal for an extensive rewrite of the current document.

At its meeting in April 1996, WG11 discussed most of the very major issues and several of the major issues. The minutes for this meeting include the resolutions taken by WG11 on those issues discussed, and requests the editor to make a number of changes to the document. In addition, WG11 delegated to the editor the responsibility for formulating responses to all remaining unresolved issues.

The next three sections of this response discuss the very major issues, the major issues, and the minor issues. The last section deals with the clause-by-clause comments.

No detailed response to the proposals for an extensive rewrite is attempted. However, it may be noted that Sweden's proposed standard and LIA-2 would serve quite different purposes: LIA-2 is intended to guarantee portability of programs with reasonable accuracy and speed, but with sufficient flexibility to allow vendors to compete in the marketplace. Sweden's primary concern appears to be to minimize the variation in the possible results a program might return. It follows that LIA-2 provides only limited support for double precision; just enough to allow some use in critical parts of an "ill-conditioned" problem. Sweden, on the other hand provides quite extensive support for multi-precision calculations.

Many of Sweden's comments deal with making LIA-2 follow the style chosen for LIA-1. This is cumbersome: The operations in LIA-2 are far more complicated than those in LIA-1, and it is hardly to be expected that the descriptions suitable for one are also suitable for the other. In fact, as Sweden has noted, more than one style is used in LIA-2.

2 The Very Major Issues

The very major issues are items 3.b, 3.c, 3.d, 5.b.i, and 7.a in the first section of Sweden's comments. The actions taken by WG11 were as follows:

****

1. Issue 3.b: Underflow and Overflow Checks: In its last resolution, WG11 instructs the editors to include the input value conditions under which the properties listed under Axioms and Exceptions hold.

Also, WG11 requests the editors to verify that the aggregate of all such conditions cover all possible combinations of input values for that operation. However, conditions involving **NaN***s* need not be specified explicitly.

The minutes of the meeting say nothing about the use of a generalized "*result_F*," as recommended by Sweden in 3.b.

> LIA-2 now includes everything requested by WG11. A generalized *result_F* is neither defined nor used.

****

2. Issue 3.c: Continuation values for underflow: Sweden covers only systems which support denormalized numbers and "$-$**0**."

> This item is not among WG11's resolutions. However, it is hard to imagine anyone having a problem with it; it is now included in LIA-2.
>
> For an implementation which does not support denormalized numbers and $-$**0**, LIA-2 requires the implementation to choose a continuation value and document its choice.

****

3. Issue 3.d: Missing Cases in the Specifications – including renaming of "undefined:" WG11 requested the editor to replace "undefined" by "bad_argument" and include documentation in Annex A and permission (in the Notification clause) for an implementation to use a common representation for the two terms.

> LIA-2 now includes the replacement and the requested documentation.
>
> The editor opposes the replacement. The problem is that LIA-1 uses "undefined" for all divisions by zero, regardless of whether the dividend is or is not zero, while in LIA-2 only the case of 0/0 is "**bad_argument**;" all other divisions by zero return "**pole**." The above resolution eliminates the term "**bad_argument**" from LIA-2 – and also from LIA-1 at some time in the future. A better resolution is to keep "**bad_argument**" in LIA-2 and, at some time in the future, add "**pole**" to LIA-1.

****

4. Issue 5.b.i: Replace sumhi and sumlo.

> Although none of the WG11 resolutions deal with this issue, WG11 appears to request that LIA-2 specify their behavior for all values in F, and suggests that the editors consider the definitions submitted by Sweden.
>
> LIA-2 now contains revised specifications for sumhi and sumlo, but does not use the definitions submitted by Sweden.
>
> The purpose of LIA-2's "doubled precision" operations is to expedite limited use of extended precision to maintain accuracy in (possibly) ill-conditioned applications. Such limited use often occurs in "inner loops" so that high performance is important. LIA-2 implicitly assumes that (in these applications) all input operands have been preprocessed so that any occurrence of a floating point exception indicates that the program is broken. See the paper "On the Orthogonality of Eigenvectors Computed by Divide-and-Conquer

Techniques" by Sorenson and Tang in the SIAM Journal of Numerical Analysis, Vol. 28, No. 6 for a description of such limited use.

It appears that the purpose of the corresponding operations in Sweden's documentation is to support accurate multi-precision, for which recommendations for other supporting operations are made. The editor's view is that such support is best provided in a separate standard for operations to arbitrary precision.

****

5. Issue 7.a: External Forms Conversions: Treat all conversions in the same manner.

There were many flaws in the specifications for the conversion operations. An effort has been made to remove the flaws.

Note that the conversions in the subclauses of clause 15 are intended to be available to programmers via high level language bindings. The conversions in the subclauses of clause 16 cover a wider variety of input formats and are intended to provide guidance to the writers of compilers and runtime software. An implementation is not required to provide such operations.

3 The Major Issues

There are fifteen major issues, as follows:

****

1. Issue 1.a.i: The symbol $-\mathbf{0}$ is ambiguous.

This is a special case of Issue 1.b; see Item 4 below.

****

2. Issue 1.a.ii: Use of Floor and Ceiling Brackets: The generalizations given in clause 4.1 should not be used.

WG11 did not discuss this issue.

The editor disagrees: these generalizations greatly ease specification of those conversion operations for which the result is a floating point number. They have not been removed from LIA-2.

****

3. Issue 1.a.iii: An unquoted infinity symbol should be used only for the mathematical infinities.

This is a special case of the next item.

****

4. Issue 1.b: IEC 559 Special Values must be distinguished from conventional mathematical values.

WG11 agreed and decided on the use of bold face for the special values $-\mathbf{0}$, $+\boldsymbol{\infty}$, $-\boldsymbol{\infty}$, and **NaN**.

This change has been made in LIA-2.

****

5. Issue 1.d: Eliminate use of the repeated F as subscripts in the names of the operations.

This issue was not discussed by WG11.

The change has not been made in LIA-2, partly because of the following:

Sweden refers to a "complete list of suggested names" given among its comments. This list involves specification of a number of intermediate operations and "helper" functions, in terms of which the final operations are specified.

The use of intermediate operations and helper functions almost requires their implementation by a vendor in order to test and guarantee conformance. This, in turn, limits the choice of algorithms to be used.

This approach is desirable for a vendor who expects his underlying structure to remain unchanged for extended periods of time. However it imposes undesirable constraints on competing vendors, who need greater flexibility in order to survive in a competitive marketplace.

The goal of LIA-2 is to provide the minimum constraints necessary to guarantee that conforming vendors can provide reasonably consistent and reliable results. The present naming convention is adequate for the limited number of operations needed to meet this goal.

****

6. Issues 2.a.i through 2.a.iv: Heading Structure: These issues propose an extensive reordering and reqrouping of the operations.

The interpretation of the resolution passed by WG11 on this issue is unclear.

The editor opposes this change. The current grouping of the operations was organized for the benefit of the writers and users of scientific software, the audience for which the standard was originally intended. The transcendental operations were the only ones originally planned for the standard. Those operations performing similar mathematical tasks were grouped together. Then the groups were ordered primarily in accordance with their expected frequency of use.

Sweden states that the proposed change is for the benefit of implementors and "readers." Some close relationships among the trigonometric functions are destroyed by this organization: e.g. Sweden's document has the specifications for the sine of a radian argument separated by almost three pages from those for the sine of an argument in arbitrary units. This is done because Sweden considers it is important to keep all "radian" operations together, etc.

The editor considers it MUCH more important to group all "sine" operations together.

****

7. Issue 3.a: Notational Style: Use "LIA-1 Definitions" Style for Integer Operations, Conversions and Non-transcendental Floating Point Operations. An important element of this issue is that "underflow/overflow detection be based on the computed result, rather than on (an inverse of) the mathematical result."

This issue was not discussed at the WG11 meeting.

None of these changes have been incorporated into LIA-2 for the following reasons:

It is more important that the style be tailored to the material in LIA-2.

In particular, LIA-2 bases the thresholds for overflow and underflow on values for the input argument in accordance with WG11's tenth resolution. Tying the thresholds to input values often exhibits characteristic features of the corresponding mathematical function.

For example, the overflow threshold for the exponential operation shows clearly how little of the full domain of input values for the mathematical function are available on the computer. This is totally obscured by tying the threshold to the computed value of the operation.

****

8. Issue 4.a: On Return of an Integer Value, Round Ties to Even; to be applied only to those operations that explicitly always return an "integer" value, either in an integer or a floating point type.

WG11 made no resolution on this issue.

This has not been done; the editor considers that rounding to nearest should be required only for operations for which such rounding is an explicit part of the definition. In general, an implementation should be free to round all other operations in accordance with the rounding style of its underlying hardware and software.

Note, however, that LIA-2 has made an exception in the case of $SQRT_F$, in order to maintain compatibility with IEC 559.

****

9. Issue 4.b: Arguments for which Higher Accuracy Should Be Required: Sweden's comments seem to indicate that this problem is largely with the "two argument" trigonometric operations. In particular, the results of arguments such as $u/4$ should have higher accuracy than other arguments.

The LIA-2 requirements are that if the results of the corresponding mathematical functions are "representable" in F, then the operations must return the "true" result.

Sweden considers that LIA-2 has no requirements if arguments such as $u/4$ are not exactly representable. This is not true. All of these operations require that the error always be within 2 ulps of the true result, which seems adequate for the general case.

****

10. Issue 5.a.i: Remove the max/min Operations with more than two Arguments.

WG11 did not support this proposal. Hence they have not been removed.

****

11. Issue 5.a.ii: Remove all Explicit Truncation Operations:

    WG11 rejected this proposal at its winter meeting, and declined to reconsider it at the spring meeting. However, Sweden has agreed to drop it if a legitimate use for truncation is found. Some examples have been submitted, but, to date, Sweden has taken no action.

    LIA-2 still includes truncation.

****

12. Issue 5.d.iii: Add Wrapping Addition and Subtraction.

    Rejected by WG11: its ninth resolution instructs the editor to neither add nor remove operations.

    LIA-2 does not include wrapping addition and subtraction.

****

13. Issue 6.b: Decrease the number of "max_arg" parameters, and perhaps rename to "big_angle."

    The fifth resolution of WG11 supports decreasing the number of "max_arg" parameters to two, but does not mention renaming them.

    LIA-2 now contains "MAX_ARG_RAD" for trigonometric operations on radian arguments, and "$MAX\_ARG(u)$" for these operations with arguments in units of u.

****

14. Issue 7.b: Remove Truncating Operations:

    This is the same as Issue 5.a.ii, discussed above in item 11.

****

15. Issue 8: Conformity Clause: Sweden has decided to postpone this issue, and hopes to work for harmonization among all three parts later.

4 The Minor Issues

There are ten minor issues as follows:

****

1. Issue 1.d: Names of Operations: use lower case for the names of the operations (as in LIA-1).

WG11 has not discussed this issue.

NOT DONE. They are in upper case in LIA-2, in order to distinguish them from mathematical functions (often with the same name) which are given in lower case.

See the discussion (dealing with another aspect of this issue) in Item 5 under major issues.

****

2. Issue 1.e: Do Not Use "G" as a Meta-variable for a Floating Point Type.

Not discussed by WG11.

LIA-2 still has G; F and G are more readable than Sweden's suggested use of F_1 and F_2.

****

3. Issue 2.a, Heading Structure; Subissues of Issue 2.a.iv:

(a) Give inverse hyperbolic operations their own heading.

They already have their own heading.

(b) Give *arctan*2 and *arccot*2 their own heading. Sweden considers that they are not inverse trigonometric operations. Sweden also proposes renaming them.

NOT DONE. LIA-2 still lists these operations with the rest of the inverse trigonometric operations, and has not changed the names. Both the names and their identification as inverse trigonometric operations have been recognized for many years in Fortran and C, the primary languages used for scientific applications.

(c) Add a subsection for angle normalization and conversion operations for user convenience.

WG11 has not discussed this issue.

NOT DONE. It is likely that users needing such operations would also want the specifications tailored to their particular applications.

(d) Do not interleave the inverse trigonometric operations in radians and unit u; rather list all radian operations together and all unit operations together, for the convenience of the reader.

NOT DONE. LIA-2 currently groups both the trigonometric and inverse trigonometric operations so that all variants of each such operation are together; see the discussion in the sixth major issue above.

****

4. Issue 2.a.v: Place the clause on "notification" before the clauses dealing with the specification format. Sweden gives no reason for this change.

This item is part of WG11's second resolution, for which the interpretation is unclear. WG11 is currently trying to resolve this issue.

NOT DONE, pending further word from WG11.

****

5. Issue 2.a.vi: Add clause 7 on the relationship with language standards.

> This issue was not explicitly discussed by WG11. However, a discussion on "conformity" and WG11's third resolution requesting additional material in Annex B is relevant.
>
> NOT DONE, but Annex B now contains the additional material requested by WG11.

****

6. Issue 2.a.vii: Renumber the clause on documentation requirements to 8.

> NOT DONE; LIA-2 did not make the preceding change, and hence has not made this one either.

****

7. Issue 3.e: Argument order should "be consistent with conventional mathematical notation or what would be such notation."

> NOT DONE. LIA-2 has made no changes in argument order: Sweden's idea of "what would be consistent notation" is often exactly the opposite of the editor's idea.

****

8. Issue 5.c: Some Operations to Rename: WG11 did not discuss any of these proposed name changes.

(a) Rename $mullo\_I$ and $mulhi\_I$: These names are confusing because the operations are not sufficiently closely related to the floating point operations with the same names.

> NOT DONE. LIA-2 still calls them $MULLO\_I$ and $MULHI\_I$. The editor considers the reason given for a change to be unconvincing.

(b) Rename $arctan2$ and $arccot2$: Since the mathematical definition is in terms of $arccos$, the use of $arctan$ and $arccot$ is inappropriate.

> NOT DONE. LIA-2 still calls them $arctan2$ and $arccot2$. In accordance with standard practice. The definitions given in LIA-2 are in terms of $arctan$ and $arccot$, rather than $arccos$.

(c) Use "convert" or perhaps "cvt" in the names of conversion operations. "(For a complete proposal see the messages on suggested LIA-2 sections.)"

> NOT DONE. Incidentally, the "complete proposal" runs to thirteen pages, in which any reference to conversions is well hidden!

(d) Use the word "rest" instead of "rem" in the remainders for division and square root, in order to avoid confusion with the use of "rem" in LIA-1.

> NOT DONE. The editor considers that the use of the suffixes in "REM_DIV" and "REM_SQRT" are sufficient to avoid any confusion.

****

9. Issue 5.d: Some operations to add: There is a total of thirteen operations as follows:

(a) Two items deal with integer division and integer remainder.

(b) The third item deals with wrapping add and subtract, dealt with above as major item 12.

(c) The fourth and fifth items deal with modulo arguments.

(d) The sixth item wants a divide predicate operation.

(e) The seventh item deals with angle normalization and conversion operations. These are discussed in minor issue 5(c) above.

(f) The last two items deal with primitive operations for the support of interval arithmetic and extended floating point range.

> All of the above operations are integrated into Sweden's complete floating point package.
>
> NOT DONE. These operations provide a level of detail which is unnecessary for the limited portability goals of LIA-2. Moreover, their implementation would impose an immense burden on conforming implementations.

****

10. Issue 6: Parameters Issues; Issue 6.a: Accuracy Parameters: Mention each one of them explicitly.

> WG11 discussed some related issues, and covered some aspects of accuracy in its seventh resolution.
>
> LIA-2 includes an accuracy specification for each operation which returns an "approximate" result. No such specification is included for those operations which must return an exact result.

5 Page-by-Page and Clause-by-Clause Comments

A number of these comments will be identified as having already been covered in one or another of the sections above on the very major, the major and/or the minor comments.

1. Page 1:

> First paragraph: Change "real elementary" into "floating point elementary":
>
> > NOT DONE – keep "real" to distinguish from "complex" later.
>
> Fifth paragraph: change "are integer" into "are values in integer":
>
> > DONE INSTEAD - changed to read "... operand values are of integer or floating point datatypes".
>
> Sixth paragraph: refer to ANSI Common lisp as well as to ISO lisp.
>
> > NOT DONE; a reference to the international standard is sufficient.

****

2. Page 2: First paragraph: refer to (some) format standards, perhaps in a note.

NOT DONE: This would serve no useful purpose.

****

3. Page 2, Second paragraph of clause 2: Delete the first "OP", and replace the second "OP" by "the value of the operation when applied to arguments."

NOT DONE. "OP" is an easily understood abbreviation for operation,

****

4. Page 2, Clause 2, point a): Replace "operation" by "value."

DONE.

****

5. Page 2, Clause 2, point b): Replace "OP conform" by "the operation conforms".

NOT DONE. See Item 3 above.

****

6. Clause 2.1: Delete entirely.

NOT DONE. WG11 rejected this proposal.

****

7. Page 3: In the display at the bottom of the page, make the following replacements: "implication" by "implication and equivalence" and (in two places) "on R" by "on reals".

DONE.

****

8. Page 4: In the top display insert e^x and x^y. (Does the overlap with the previous display make a problem?)

e^x and x^y are added. The editor sees no problem with overlap.

****

9. Page 4, Fourth paragraph: Use F_1 and F_2 instead f F and G when two floating point datatypes are needed.

NOT DONE; F and G are more readily distinguished than F_1 and F_2.

****

10. Page 4, middle of the page etc.: Rename "arg_too_big" to "angle_too_big".

NOT DONE; the use of the word "argument" for trigonometric operations has been conventional for many years.

****

11. Page 4, Note 1: Delete and instead put quotes around the special values of IEC 559.

NOT DONE. The note provides a useful explanation and has not been deleted.

The special values are in bold face instead of being enclosed in quotes.

****

12. Page 4: Delete the material on the datatype Sequence.

NOT DONE. WG11's seventh resolution instructs the editor to neither add nor remove operations. The datatype Sequence is used in the specifications for the conversion operations.

****

13. Page 5, top line: add "and $j \neq 0$" (otherwise $0|0$ is true which is a bad idea, since 0 is not a divisor of anything, not even 0).

NOT DONE. There is a difference between "$0|x$" and "$x/0$". The former does not imply an evaluation, and is defined in textbooks as it appears in LIA-2. The latter implies an evaluation and the result of dividing x by 0 must be specially specified.

****

14. Page 5: Do not use conventional notation in an unconventional sense. In particular, do not generalize the floor and ceiling brackets.

LIA-2 still uses the generalized floor and ceiling brackets to expedite the definition of the datatype sequence, as discussed above in Item 2 of the major issues (Sweden's issue 1.a.ii).

****

15. Page 5, middle of the page: Delete the sentence starting "Predicates ... " – it is false.

NOT DONE; What is wrong with the sentence?

****

16. Page 5, Clause 4.2: Delete the numbering.

NOT DONE; WG11 authorized the numbering at an earlier meeting in response to a request from an NB.

****

17. Page 5, Clause 4.2, Item 5, Continuation Value: Replace "exception" by "notification".

    DONE.

****

18. Page 5, Clause 4.2: Rework and simplify the definitions referring to "monotonic".

    NOT DONE. No justification is given. There is no indication of what is wanted.

****

19. Page 8: In the last paragraph of "Signature" replace "input range" and "output range" by "domain" and "range," respectively.

    NOT DONE; "input" and "output" are more readily understood.

****

20. Page 8: "ulp" is used in LIA-1 and should perhaps be added to clause 4.2 of LIA-1.

    NOTHING DONE. This concerns LIA-1, not LIA-2.

****

21. Page 8: The first paragraph of clause 5.2 is carelessly worded. It takes all components together to define the operation.

    NOTHING DONE. A statement similar to the second statement above already appears in the last paragraph of clause 5. However, clause 5 has now been rewritten for greater clarity.

****

22. Page 8: The meanings of the words "definition" and "axiom" are distorted. Follow LIA-1.

    NOTHING DONE. It is unclear what Sweden thinks is wrong and how it should be fixed.

****

23. Page 8: Replace "the operation OP" by "an operation".

    NOT DONE; see Item 3 above.

****

24. Page 8, Item b): Revise the specification for the max_error parameters.

It appears that Sweden wants this parameter defined more in terms of the true result of an operation than on the computed value.

NOT DONE. The definition currently given is exactly what was intended. It allows a program to estimate the difference between the (available) calculated value of the operation and the true value, which is usually not available.

****

25. Page 9, Point c-d: specify each parameter explicitly (later on). Eliminate *error_limit_op*.

NOT DONE. The parameter *error_limit_op* has a value specified by LIA-2. This value specifies an upper bound for *max_err_op*, which has a value to be documented by the implementation, and which is dependent on the algorithm used.

****

26. Page 9, Clause 5.3: Second paragraph: Move specifications for principal ranges to clause 4.

NOT DONE. This would conflict with WG11's tenth resolution that project editors annotate each of the "axiom" and "exception" specifications to include the relevant input conditions.

****

27. Page 9, Third paragraph: "An axiom shall hold......": How does that affect the use of a constant such as "$ln(fmin_N)$" which is not in F.

FIXED. Constants such as "$ln(fmin_N)$" have been replaced by their bounding values in F. This is documented in clause 5.

****

28. Page 9, Clause 5.3 Last Paragraph: Always use side conditions for axioms, definitions, etc.

DONE.

****

29. Page 10, Clause 5.6: Replace "of integer" by "of an integer" in the last paragraph.

DONE.

****

30. Page 10, Clause 6: Replace the third paragraph to remove technical mistakes.

NOT DONE. There are no mistakes; Sweden has misinterpreted the paragraph.

****

31. Page 10, Clause 6: Expand "$+-$" in the fourth paragraph.

NOT DONE. That would make a very cumbersome paragraph.

****

32. Page 11, Second to the last paragraph: Expand "+−" and replace "*fmin*" by "$fmin_N$."

First, "+−" has not been expanded.

However, *fmin* has been replaced by $fmin_N$

But note that these changes are erroneously listed under page 10 in Sweden's notes.

****

33. Page 11, Last Paragraph: The continuation value on "*angle_too_big*" should not be a **NaN**.

The last paragraph mentions **NaN** as just one possibility.

****

34. Page 12, Point e: There should be only one or two "*max_angle*" parameters.

DONE. The two parameters are named "*max_arg_rad*" (for radian arguments) and "$max\_arg(u)$" (for arguments in other units).

****

35. Page 13, etc.: Now there are three specification styles. The LIA-1 style is best.

NO CHANGE. The various operations are often best specified in different ways.

****

36. Page 13, etc.: All individual operation headings: The names of the operations are in italics, not bold, including the subscript.

NO CHANGE MADE. Why is this a problem?

****

37. Page 13, Clauses 8.1 and 8.2: The "axiom" parts are superfluous. If needed, state in clause 4 that the mathematical sqrt function has a positive value.

The axioms essentially define the operations. Each operation contains its defining axioms.

****

38. MANY PLACES: Side conditions are missing. Fill them in.

DONE.

****

39. Page 15, $POWER\_F$: Isn't $POWER\_F(0, y)$ a pole when $y < 0$?

    YES; This has been fixed.

****

40. Page 15, $POWER\_F$: Why isn't $POWER\_F(x, y)$ undefined for $x < 0$ or $x = -\infty$?

    DONE; These errors have been corrected.

****

41. Page 15, "$ln(fmin_N)$": Replace by "$ln(fmin)$."

    NOT DONE; in context, $ln(fmin_N)$ is the underflow threshold.

****

42. Page 15, Various logarithmic operations: The use of "+infinity" and "infinity" is confusing. Clean it up.

    DONE – by use of floor and ceiling operations.

****

43. Page 18, Clause 10.3: LOG_FF is incomplete and incorrect. See accompanying document for a much more correct version.

    The specifications have been corrected, but not along the lines of Sweden's document.

****

44. Page 19, etc. Clause 11, Second Paragraph: Arcminutes and arcseconds are common units which should be in T too.

    NOT DONE. There is insufficient demand for special routines for these units. However, of course, an implementation is free to provide them.

****

45. Page 19, Clause 11, Third Paragraph: The condition on u should be $u >= r^{emin+p-1}$.

    DONE.

****

46. Page 19, Clause 11, Fourth Paragraph: This item has several parts:

    (a) The reason for the presence of the big angle parameter and angle_too_big notification is because of the "sparsity" of values for large angles. This reason should be documented in both the normative text and the rationale.

The above is one of two reasons for the big angle parameter. Sweden overlooks the fact that, at least in the past, there have been implementations for which argument reduction is inaccurate for sufficiently large (but representable) arguments. LIA-2 does not wish to forbid such implementations which sometimes serve a useful purpose. However, these parameters should reflect any such inaccuracies in the vendor's implementation of the trigonometric operations.

(b) Rename "*arg_too_big*" to "*angle_too_big*" because LIA-2 uses it only in connection with angles.

NOT DONE. That change might necessitate invention of another parameter in the future.

(c) Rename the "*max_arg*" parameter(s) to "*big_angle*" since it is not maximal and refers only to angles.

NOT DONE – for the same reason as in the previous item. Moreover, the parameter is "maximal" in the context of the operation to which it refers.

(d) There should be only one or two "big_angle" parameters, defined by LIA-2.

DONE. See item 34 above.

(e) Angle_too_big notifications should be handled via recording of indicators, unless there is an explicit request to the contrary.

NOT DONE. LIA-1 does not require an implementation to support recording of indicators.

(f) Do not make the big_angle parameters into functions. It suffices to divide the argument by the unit.

NOT CHANGED: experience with unit argument operations is insufficient to predict what their needs will be.

****

47. Page 20, etc., Clause 11 Operations: This item has several parts:

(a) In the specifications for the unit argument trigonometric operations, u is first defined to be in F, and then allowed not to be in F. (The top line declaration covers the Extensions too.)

The contradictions have been removed. It is now noted in clause 5 that the top line does not refer to the Extensions, which contain their own statements on u.

(b) Two maximum error parameters are needed for the unit argument operations: one for units in T and another for units not in T. The first of these should be the same parameter as for the corresponding radians operation.

This has been fixed. The parameter "$max\_err(u)$" serves this purpose because its value depends on that of u.

It is unclear why "max_err_rad" should be the same parameter as "$max\_err(u)$", for $u \in T$.

(c) Current Clause 11.1: Find a wider interval for the second axiom. Similarly for 11.4, 11.9 and 11.10.

DONE. The interval is now $|x| < r^{-p/2}$.

(d) Insert an axiom for small arguments in Clause 11.3.

DONE.

(e) Remove the spurious end parenthesis in first axiom of 11.5.

DONE. The interval is now $|x| < r^{-p/2}$.

****

48. Page 20 Clause 11.6: Either define the notation "...= =...(mod)" and use it for all periodic operations, or replace it.

DONE; "mod" is no longer used.

****

49. Clause 11.13: The signatures for the degree operations are missing.

Signatures have been added. See item 3(d) of the minor issues.

****

50. Page 26, etc., Clause 12: This item has three parts:

(a) The principal value ranges should be given in Clause 4. They refer to mathematical functions and do not make sense for numerical operations.

NOT DONE; see the discussion in Item 26 on clause 5.3 above.

(b) Replace $u = 0$ by $u \geq r^{emin+p-1}$.

DONE.

(c) Rounding to nearest must be allowed and encouraged for mathematical constants entering specifications on results of an operation.

NOTHING DONE; An implementation is free to choose the circumstances under which rounding to nearest is used (except for those operations whose definitions require rounding to nearest).

****

51. Unit Argument Inverse Trigonometric Operations: This Item has three parts:

(a) The first part is the same as Item 47(a) above, except it refers to the inverse (instead of the direct) operations.

The response is the same as in Item 47(a).

(b) The second part states that the Extensions components are incomplete for these and many other operations.

The Extensions components have been completed.

(c) The third part is the same as Item 47(b) above, except it refers to the inverse (instead of the direct) operations.

The response is the same as in Item 47(b).

****

52. Clauses 12.1 and 12.5: Find a wider valid interval for the second axiom.

DONE.

****

53. The Arc/Angle/Phase Operations: This item is in three parts:

(a) Join the "*arctan*2" and "*arccot*2" operations.

NOT DONE, because of potential language incompatibilities.

(b) Rename them.

NOT DONE. The present names have been in common use for many years.

(c) Give them their own clause, between "trigonometric operations" and "inverse trigonometric operations."

NOT DONE; They ARE inverse trigonometric operations.

****

54. The *arctan_F* and *arccot_F* operations have a jump at zero, consistent with Abramowicz but not with Ada. LIA-2 should not take a stand but specify both.

NOT DONE; LIA-2 follows Abramowicz and Stegun in case of any conflict, and will continue this policy in the future.

****

55. These same two operations may underflow, a possibility not currently mentioned.

arctan_F contains underflow, correctly specified. *arccot_F* did not – this has been fixed.

****

56. Possible Underflow in Clauses 12.*: For radian operations, clauses 12.1, 12.5, and 12.9. For unit argument operations, clauses 12.2, 12.4, 12.6, 12.10, 12.14, and 12.16.

FIXED; underflow is now correctly specified in all of these operations.

****

57. Clause 12.7: In four parts:

(a) Define the mathematical angle/arc function in terms of inverse cosine.

NOT DONE. This function is almost universally defined in terms of inverse tangent.

(b) The results in the table are not in F, and hence are not required.

This has been fixed for all the inverse trigonometric operations, see Items 47(a) and 51(a).

(c) The underflow criterion is incorrect (x must be positive for underflow but the sign of y does not matter).

DONE; the error has been corrected.

(d) Sweden does not like the proliferation of specification styles.

NOTHING DONE; LIA-2 chooses specification styles so as to achieve completeness and correctness.

****

58. Clause 12.8: In three parts:

(a) The results in the table might not be in F.

NOTHING DONE; see Items 47(a) and 51(a) above.

(b) The underflow criterion is incorrect.

DONE; this error has been corrected.

(c) The signatures for the degree operations are missing.

FIXED; they are included in the current specification; see Item 49 above.

****

59. All of the Trigonometric and Inverse Trigonometric Operations: Group them as trig. ops, arc/angle ops, and inv. trig. ops. Also do not interleave the radians and unit arg. ops. Instead order them as follows: radian ops. first, then unit arg. ops. second, then degree ops. third.

NOT DONE; see Item 6 of Major issues (Sweden's issue 2.a.iv). above.

****

60. Page 33, etc., Clauses 13 and 14: In many parts:

(a) Clauses 13.1, 13.3, 14.1, and 14.3 may underflow (for subnormal non-zero input). Specify intervals (around 0) for which return of the argument provides the most accurate result. Also examine trigonometric operations.

DONE.

(b) Clauses 13.2 and 13.5: Specify intervals (around 0) for which 1 is the most accurate output, and examine some trigonometric operations.

DONE.

(c) Clauses 13.3 and 13.4: Specify intervals (far from 0) for which 1 is the most accurate output, and examine some trigonometric operations.

DONE.

(d) Clause 13.4: *coth* may overflow.

FIXED.

(e) Clause 13.6: $csch(-infinity) = -\mathbf{0}$.

DONE.

(f) Clause 14.1: The sign requirement is missing.

FIXED.

(g) Clause 14.1: Use of $ln(2 * fmax)$ may disallow round-to-nearest.

Floor and ceiling functions are now used; see Item 41 above.

(h) Clause 14.2: This item has three parts:

(1) The smallest result for $arccosh\_F$ is 0 (not 1).

Correction made.

(2) $arccosh\_F$ is undefined for all input < 1; remove absolute value signs.

DONE.

(3) The "$\leq ln(2 * fmax)$ requirement" may disallow round to nearest.

Floor is now used.

(i) Clause 14.4: This operation may underflow.

Specifications for underflow have been added.

(j) Clause 14.5: $Arcsech\_f(-0) = pole(+\infty)$

CORRECTION MADE.

(k) Clause 14.6: This item has two parts.

(1) $Arcsech\_F$ may underflow for negative arguments.

(2) $Arcsech(-\infty) = -\mathbf{0}$

Both are FIXED.

****

61. Page 37, etc.; Clauses 15 and 16: This item has many parts:

(a) Clauses 15.1, 15.2, 15.3: Delete "undefined" from the signature; these operations are defined for all arguments in F.

DONE.

(b) Clause 15.4: Delete this operation which has no raison d'etre.

NOT DONE; the operation is included in languages such as C and Fortran.

(c) Clauses 15.5, 15.6, and 15.7: These clauses are not conversions, so they should not be so classfied.

NOTHING DONE; they are "conversions" of the input operands.

(d) Clause 15.8: Delete this operation which has no raison d'etre.

NOT DONE; the operation is included in languages such as C and Fortran.

(e) Clauses 15.9, 15.10, and 15.11: This item has three parts:

(1) Specify in terms of a "*result_F*" function, as in LIA-1.

NOT DONE; consistency within LIA-2 is more important than with LIA-1.

(2) These operations may result in underflow.

Handling of underflow has been added.

(3) The specifications for the extensions are incomplete. They do not include all combinations of input for the source and target operands.

FIXED; the extensions are now complete.

(f) Clause 15.11: The continuation value for overflow is inconsistent with IEC 559.

FIXED.

(g) Clause 15.12: Delete this operation which has no raison d'etre.

NOT DONE; the operation is included in languages such as C and Fortran.

(h) Clauses 16, 16.1 and 16.2 should be tightly joined with the operations in Clause 15. There is no reason for a strong separation.

NOT DONE; these two classes of operations serve different purposes. See very major issue 5.

****

62. Clauses 17.1 - 17.4: Either exclude all operations on sequences, or include "all" primitive binary arithmetic operations generalised to sequences.

NOT DONE. WG11 has decided to keep only the max/min operations on sequences; these operations have been part of Fortran for many years.

****

63. Clauses 17.5 -17.8: The axioms are superfluous and should be deleted.

NOT DONE; explicitly including commutativity is useful.

****

64. Clause 18: This item has four parts.

(a) The doubled precision operations are useful for more than "doubled precision."

NO CHANGE MADE. Considerably more elaborate specifications are needed to support "multi-precision," which appears to be what Sweden has in mind.

(b) The error limits for *sublo_F* and *addlo_F* are strange.

The error limits would be strange for full support of multi-precision. They are good enough for doubled precision.

(c) If rounding is to nearest then the result for these operations can be exact, with no error allowed.

NO CHANGE MADE; this is more stringent than required for doubled precision, and would cost performance.

(d) The current specifications are incomplete.

NO CHANGE MADE; they are complete enough for their present purpose.

****

65. Page 48, Clauses 18.2.3, 18.2.4, and 18.2.5: This item has five parts:

(a) Clause 18.2.3, *mullo*: This operation will not normally return an exact result when the result is subnormal.

Exactness of results is unimportant for doubled precision. Furthermore,it is unclear that the current specifications imply anything about exactness of results.

(b) Clause 18.2.3, *mullo*: The extensions portion is incomplete.

The extensions portion has been rewritten for this and almost all other floating point operations.

(c) Clause 18.2.4, *rem_div_F*: The extensions portion is incomplete and incorrect; zero divisors are not excluded.

The extensions portion has been rewritten and corrected.

(d) Clause 18.2.4, *rem_div_F*: This operation does not usually return an exact result when the result is subnormal.

Exactness of results is unimportant for doubled precision. Furthermore,it is unclear that the current specifications imply anything about exactness of results.

(e) Clause 18.2.5, *rem_sqrt_F*: This operation can underflow for certain arguments.

The specification has been revised to include underflow.

****

66. Clauses 18.3.1 and 18.3.2, *sumhi_F* and *sumlo_F*: These are strange operations and must be replaced entirely (with *add3_F* and *add3_mid_F*.

The specifications for *sumhi_F* and *sumlo_F* have been extensively rewritten They have not been replaced by *add3_F* and *add3_mid_F*.

****

67. Clause 19.1, $dprod\_F \rightarrow G$: This item is in three parts:

(a) (*dprod* would be better named $mul\_F \rightarrow F$) It would be easier to allow this operation to be defined for any two different floating point types.

NO CHANGE MADE. The operation is primarily useful for narrower to wider types. Moreover, it is specifically included to support a relatively recent extension to Fortran.

(b) For some operands $+infinity$ is returned. But even if those values are in the source type, they might not be in the target type.

The text in the extensions component (clause 5.6) has been revised to resolve this problem.

(c) The extensions portion is incomplete.

The extensions component is now complete.

****

68. Clause 19.2, mul_add_F: The overflow and underflow conditions should be revised.

DONE.

****

69. Clause 19.2, mul_add_F: The extensions portion is incomplete and inconsistent with multiplication and addition in IEC 559.

The extensions component is now complete. The specifications are adequate for doubled precision, but not for full support of multi-precision.

****

70. Clause 20.1, $hypot\_F$: May underflow for certain arguments.

The specifications require that any occurrence of underflow be "made invisible."

****

71. Clause 20.1, $hypot\_F$: The extensions portion is incomplete.

FIXED.

****

72. Clause 20.3, dim_F: The overflow and underflow conditions should be revised.

NOT DONE. What is wrong with them?

****

73. Clause 20.4, GCD_I: This item is in many parts:

    (a) Extend the specifications to include integer overflow.

    DONE.

    (b) Delete "ninZ."

    "ninZ" is replaced by "n in Z."

    (c) Delete the axioms part; it is superfluous.

    NOT DONE; the statement of axioms defines the operation.

    (d) Delete "$v \geq 1$" It is superfluous.

    NOT DONE; it does no harm and might avoid questions later.

    (e) The "," should be an explicit "and".

    DONE.

    (f) The side condition "if $x \neq 0$ or $y \neq 0$" should be written out explicitly.

    NOTHING DONE; the quoted phrase does not occur in clause 20.4.

****

74. Clause 20.5, LCM_I: The ","s should be explicit "and"s.

    DONE.

****

75. Clause 20.5, LCM_I: Delete the continuation value for integer overflow.

    NOT DONE; its presence contributes to portability of programs, even if integer overflow occurs.

****

76. Page 53, Rationale: The third sentence is not true (yet).

    We hope it is true now.

****

77. Clause A.1, Scope: Replace "his" by "their" – or reword the sentence.

    NOT DONE; "his" refers to "vendor" which is in the singular.

****

78. Clause A.1.2: Replace "standardization" by "standardisation."

    DONE.

****

79. Clauses A.2, A.3, A.4: It looks strange not having any text under some headings.

    This is done in order to keep the clauses similarly numbered in the standard and in the rationale.

****

80. Clause A.4.1, First paragraph: The sequence types are used only by operations that should be deleted.

    BUT WG11 has decided to keep them

****

81. Clause A.4.1, Second paragraph: The conversion operation referred to is no longer in LIA-2.

    Reference to this operation has been removed.

****

82. Clause A.5.1, Signature: Delete the second paragraph.

    NOT DONE; but it has been reworded for clarity.

****

83. Clause A.5.2, First paragraph: Nobody has ever (as far as I know) asked for such specifications.

    Similar statements have appeared in "older" math libraries. This paragraph is intended to discourage continuation of such practices.

****

84. Clause A.5.2, Second paragraph: The error formula is flawed. Adapt the formula used in LIA-1.

    NOT DONE; See Item 24 above. LIA-1 and LIA-2 have different requirements. It was important in LIA-1 to maintain compatibility with the corresponding operations in IEC 559. There is no such problem for LIA-2 because the only operation in common is SQRT, which follows IEC 559. The error bounds for all other operations in LIA-2 are defined in terms of the computed values of the operations. This means that they can be used in a program to estimate the errors produced by the operations invoked.

****

85. Clause A.5.3: Replace "may exist" by "are".

    DONE.

****

86. Clause A.5.6.2: Replace "+- infinity" by "+infinity or -infinity".

    DONE.

****

87. Top of Page 58, Items b) and c): Mathematical and IEC 559 infinities have been confused, and $-\mathbf{0}$ is treated inappropriately.

    They have been rewritten.

****

88. Last sentence of A.5.6.4: Use italic font for "x".

    DONE.

****

89. Last sentence of A.5.6.4: Add "unless the function is undefined via that approach, in which case the result for -0 is the same as for 0."

    DONE, with more concise wording.

****

90. Clause A.5.7: Exact mathematical expressions (2^x and 10^x) are confused with approximate operations.

    FIXED.

****

91. Clause A.5.7: EXP_F(x*ln(2)) is not a properly formulated specification.

    FIXED.

****

92. Clause A.8, Second Paragraph: "I" has no "p" parameter.

    That entire paragraph is suitable for $SQRT\_F$, and has been moved (with appropriate changes to clause A.8.1). A properly modified paragraph has been inserted in clause A.8.2.

****

93. Clause A.8.3: Really?

    An explanatory paragraph has been added.

****

94. Clause A.9, Second Paragraph: Exact mathematical expressions are confused with approximate operations.

FIXED.

****

95. Clause A.9.1: "+-infinity" should be "+infinity or -infinity".

DONE.

****

96. Clause A.9.3: An exact mathematical expression has been confused with an approximate operation.

FIXED.

****

97. Clause A.9.3, Fourth Paragraph: "k" should be in italics.

DONE.

****

98. Clause A.9.3, Fourth Paragraph: Consider IEC 559 infinity inputs to multi-argument operations.

Some additional discussion has been added.

****

99. Clause A.9.3, Fifth Paragraph: The equation is NOT a true identity. Does r mean the radix parameter?

$POWER\_FF(x, y)$ has been replaced by x^y. Yes, r means the radix parameter.

****

100. Clause A.9.5: Delete the last sentence! Heeding it would lose accuracy which is not acceptable.

The last sentence has been revised to note loss of accuracy and "unacceptability."

****

101. Clause A.11, Trigonometric Operations: This item has many parts:

(a) Second Paragraph: Replace "size" by "magnitude."

DONE.

(b) Second Paragraph: Delete "a quarter or".

NOT DONE; this (informative) paragraph describes some (more or less) common practices; it is not a specification.

(c) Second Paragraph: The last three sentences are false; delete them

NOT DONE; these sentences describe some (more or less) common practices; they are not specifications.

(d) Third Paragraph: Some implementations use an approximate value of π.

NO CHANGE MADE; the point of this statement is that n digits for π imply no more than n digit accuracy for the reduction.

(e) Fourth Paragraph: Make a clearer distinction between the radians case (necessarily approximate argument reduction) and the unit argument case (always exact argument reduction).

Some rewording has been done to avoid the many misunderstandings made by Sweden. The whole point of Clause A.11 is that LIA-2 does not require the maximum possible accuracy for the trigonometric operations. In particular the accuracy of argument reduction is left to the discretion of the implementor. Instead, this clause discusses some of the variations which have occurred in the past.

(f) Fifth Paragraph: Replace "sin" by "sec".

DONE.

(g) Sixth Paragraph: Replace "two" by "unit".

DONE; NOTE that this item should have referred to the ninth paragraph.

(h) Seventh Paragraph: "SIN" (etc.)? The radians versions or the unit arguments versions or both?

It refers to the versions with units arguments. This paragraph is the EIGHTH paragraph of Clause A.11.

(i) Ninth Paragraph: Remove the last sentence, or say it in a less "negative" way.

The wording in the comment has been adopted.

****

102. Clause A.11.6: Put the two occurrences of "u" in italics.

DONE.

****

103. Clause A.11.7: Replace "are ±infinity" by "is $+\infty$ or $-\infty$."

Rewritten to eliminate "±infinity".

****

104. Clause A.11.7: Replace "arguments of ±0, respectively" by "an argument of 0 or $-\mathbf{0}$".

Rewritten to eliminate "± 0".

****

105. Clause A.11.9: Put sec(x) in italics.

DONE.

****

106. Clause A.11.9: Replace "they" by "such arguments".

DONE.

****

107. Clause A.11.11: Replace "are $\pm$infinity" by "is $+\infty$ or $-\infty$."

Rewritten to eliminate "$\pm$infinity".

****

108. Clause A.11.11: Replace "± 0, respectively" by "0 or **-0**".

Rewritten to eliminate "± 0".

****

109. Clause A.11.13: LIA-2 should not allow different results from the degree versions and the unit argument versions.

That is not intended. This clause has been eliminated; its material is now in clause 11, and the wording has been clarified.

****

110. Clause A.12: Replace "The undefined notification is the only one" by "The undefined and underflow notifications are the only notifications".

DONE.

****

111. Clause A.12.7: This item is in several parts:

(a) Second Paragraph: Replace "the value 1" by "the value 0".

NOT DONE; this is a quote from the references.

(b) ARCTAN2: Does this mean a mathematical function arctan2; such has not been defined in LIA-2.

The wording has been clarified.

(c) Second paragraph: replace “as it approaches” by “to approach”.

DONE.

(d) Third Paragraph: Replace “±infinity” by “$+\infty$ and $-\infty$”.

Reworded to eliminate “±infinity”.

(e) The Table: Results listed cannot be returned because they are not in F.

The table is not a specification; it is a quote from another document.

(f) In the Table: Put b in italics.

DONE.

(g) Last Paragraph: Expand “$\pm\pi/4$” and “$\pm3\pi/4$”. Also replace “3π” by “$3*\pi$”.

DONE.

****

112. Clause A.13.1: Add “or greater” to the end of the first sentence.

DONE.

****

113. Clause A.13.1: The second sentence is not supported by the normative text.

Second Sentence is Removed.

****

114. Clause A.13.2: The second sentence is not supported by the normative text.

Second Sentence is Removed.

****

115. Clauses A.14.3 and A.14.4: Expand “±”.

DONE.

****

116. A.15 Floating Point Conversion Operations: Replace “a floating point datatype” by “a single value of a floating point datatype.”

NOT DONE; it reads OK as it stands.

****

117. Clause 15: Replace “integer types and floating point types” by “another integer type and to a floating point type.”

NOT DONE; the proposed replacement is more cumbersome than the original.

****

118. Clauses A.15.1 - A.15.4 and A.15.5 to A.15.8: Very repetitive and boring.

The text has been revised and improved.

****

119. Clauses A.15.9 - A.15.12: Very repetitive and boring, and also false. They contradict the specifications in 15.9 - 15.12.

The false and contradictory statements have been corrected, and the text has been revised and improved. The problem was that there was not time to change these clauses to match changes made in the standard.

****

120. Page 69, Clause A.18: There are (no longer) operations addhi, mulhl, and divhi.

NO CHANGE; clause A.18 doesn't say there are.

****

121. Clauses A.18.3.1 and A.18.3.2 (pages 71 and 72): The operations sumhi and sumlo should be replaced by different operations.

The sumhi and sumlo operations have been revised, but not along the lines suggested by Sweden.

****

122. Clause A.20.1 *hypot_F*: The statement "it can never produce an underflow" is false. For most $|x|$ with $|y| < \mathit{fmin}_N$ it shall underflow.

The specifications for *hypot_F* require an implementor to suppress any underflow notification which might occur. In this sense, the operation *hypot_F* will never produce an underflow.

****

123. Annex B: Needs updating; some operations have been removed, and some new ones added. (Annex B should be expanded into a sample bindings annex.)

Annex B has been extensively revised. However, time has not permitted the accumulation of much of the data needed to complete Annex B.

****

124. Annex C: The following changes are needed:

(a) Insert “Standards” between “International” and “Documents”.

(b) Replace “National and Other Documents” by “National Standards Documents”.

(c) Replace “Books and Articles” by “Books, Articles, and Other Documents”.

(d) Item [6]: Refer to Ada95 instead.

(e) Item [14]: 1994..

(f) Item [24]: Insert a comma after Goldberg and capitalize arithmetic.

(g) Delete the period in “W.”

(h) Delete the period in “W.” and capitalize microsystems.

DONE; all of them.

Response to the United Kingdom's Voting Position on LIA-2

The United Kingdom voted NO to CD approval, however this would become a YES vote if item 1 below is actioned and the requested Annex produced.

Summaries of the UK comments, together with responses, follow.

****

1. Create a new normative IEC 559 annex to include all definitions and references to IEC 559. This would localize the use of the special symbols, and clearly distinguish them from conventional mathematical values, for the benefit of users.

    NOT DONE. The IEC 559 special values are now an integral part of most current and new implementations. However, they now appear in bold face type to indicate their "specialness."

In any event the heading "Extensions:" should probably be "IEC 559 special values:" to emphasise the meaning.

    DONE.

****

2. In a large number of cases the Extensions sections are very difficult to understand. It would add clarity if

a) the cases were enumerated without If/Else
b) all unspecified cases were defined in a common section to have "undefined" values, thereby eliminating many entries.

    DONE.

****

3. In 1.1 para 2 insert "values" after "operands"

    DONE.

****

4. The first sentence of the second paragraph of clause 1.1 defines a value set which does not include 16.1.2, 16.1.3, 16.2.1, 17.1.4 and a number of other places. Appropriate wording needs to be used.

    NOTHING DONE: The meaning of the comment is unclear.

****

5. In 4.1 add *sub_I* to the list taken from 10967-1 (referenced in 20.2).

    DONE.

****

6. In 4.1 in the definition of Sequence change S to s (3 times)

   DONE.

****

7. In many definitions there are some conditions on the right hand side of the page as well as on the left. All conditions should be incorporated into a consistent set, in the left hand side format.

   This has been done, except that all conditions are now on the right hand side of the page.

****

8. In 4.2 change definition 1 to "discrepancy between a computed floating point result and the corresponding true mathematical result"

   DONE.

****

9. In 4.2 NOTE 1 change last sentence to "Complex number datatype requirements will be defined (specified?) in Part 3..."

   DONE – using "specified" instead of "defined."

****

10. In 4.2 definition 6 delete "a" twice

   DONE.

****

11. In 4.2 definition 7 delete "The" and make second sentence a NOTE or EXAMPLE

   DONE – the second sentence is now an example.

****

12. In 4.2 definition 8 make the brackets a NOTE

   DONE.

****

13. In definitions 8 and 9 of clause 4.2, the usage/definition of "exception" doesn't conform to other usage of the term.

   This text is copied from LIA-1, for informative purposes.

****

14. In definitions 28, 29, and 30 of clause 4.2, eliminate the word "selects" and use the appropriate inequality instead.

a) $rnd(u) \leq u$ is used in definition 28.

b) $rnd(u) \geq u$ is used in definition 29.

c) $|rnd(u)| \leq |u|$ is used in definition 30.

DONE.

****

15. In 4.2 make the brackets in definitions 31 and 32 a NOTE.

DONE.

****

16. The NOTE in 5.9 should be an EXAMPLE.

There is no clause 5.9.

****

17. In 20.1 HYPOTf(+inf,+inf) = **floating_overflow** would be consistent with the main definition, alternatively if IEC 559 special values are available for use then the main definition exception should probably change to produce $+\infty$ rather than **floating_overflow** (or possibly **floating_overflow** defined to be $pinf$).

This would apply to many other cases and support the production of a specific Annex related to IEC 559.

LIA-2 assumes that if infinities are available for input, they are also available for output. Moreover, if they are available, LIA-2 uses them wherever appropriate.

****

18. In 20.3 in the second definition replace $x - y$ with $sub\_F(x, y)$ or other appropriate function since the mathematical exact value $x - y$ is not always representable.

DONE.

Response to the United States' Comments on LIA-2

In April 1996, the United States voted as follows on LIA-2:

1.The U.S. supports the registration of CD 10967-2.

2. U.S. National Body votes to Disapprove ISO/IEC CD 10967-2, see attachment for comments.

Attached Comments:

The following (primarily editorial) changes should be made to ISO CD 10967-2 before further progression.

As indicated below, almost all of these changes have been made.

****

Clause 4.1: This item has five parts:

First sentence of second paragraph: Reword to provide for the use of $+$ and $-$ as parts of the symbols for -0, and the signed infinities of IEC 559.

DONE – in the last paragraph.

In the first line of the notations used, add the words "and equivalence."

DONE.

In the third and fourth lines replace the symbol RR by the word "reals."

DONE.

Insert "e^x, x^y" as the first line in the list of ideal mathematical functions defined in the cited reference.

DONE.

Revise Note 2, if necessary.

NOT NEEDED.

****

Clause 5 and its subclauses: Note that all components of the specifications, except the extensions component, apply only to input operands in F and I. Only the extensions component deals with inputs of $-\mathbf{0}$, signed infinities (and $\mathbf{NaN}s$).

DONE – in clause 5.

Decide whether constraints in definitions and axioms involving mathematical quantities such as $ln(fmax)$ must be literally met or whether violation by a rounding error is permitted. Then document this decision in the appropriate place.

It has been decided that constraints in definitions and axioms should refer to rounded values. This decision is documented in Clause 5.

****

Clause 5.2: Note that the Definition component applies only to input operands in F.

DONE.

****

Clause 5.3: Note that the Axioms component applies only to input operands in F.

DONE.

****

Clause 5.4: Note that the Exceptions component applies only to input operands in F.

DONE.

****

Clause 5.6: In the second line, replace "see 6" by "see clause 6".

DONE.

In the last line, replace "of integer datatype" by "of an integer datatype."

DONE.

****

Clause 6: In the last line of the fourth paragraph, replace "$\pm$infinity" by "$+\infty$, $-\infty$,".

DONE.

****

Clause 8.1: Change the title to "Floating square root".

DONE.

Add "if $x \geq 0$" to the axiom. Add "if $x < 0$" to the exception.

DONE – both are added.

****

Clause 8.2: Add "if $x \geq 0$" to the axiom.

DONE.

****

Clause 8.3: Add "if $x > 0$" to first axiom.

DONE.

****

Clause 9.1: Replace second axiom by "$EXP\_F(x) = 1$ if $|x| < r^{-p}$."

DONE.

****

Clause 9.2: Delete the second axiom.

NO – it is obvious, but it does no harm.

****

Clause 9.3: In first exception insert $+\infty$ as a continuation value.

DONE.

In fifth row of the table, replace the second and third entries by dashes.

DONE.

****

Clause 10.1: Add "if $x > 0$" to first axiom.

DONE.

****

Clause 11: First, decide whether LIA-2 should or should not include the "two-argument" trig operations. Then, if they are to be eliminated, what about the "degree" operations? they could just fill the space currently occupied by the two argument operations. The editor recommends that LIA-2 eliminate the two-argument operations, and replace them by full specifications for the degree operations.

If the two argument operations are to be kept, Sweden recommends that the lower limit of 0 for u be replaced by $r\_F^{emin\_F+p\_F-1}$.

The two argument operations have been kept, using Sweden's lower limit for u.

****

Clause 11.1: Replace $fmin_N$ by $r^{-p/2}$.

NOT DONE; that axiom has been eliminated.

****

Clause 11.2: Add another axiom: $SIN\_FF(0, u) = 0$.

DONE.

Replace ") <" by ")| <" in the last exception.

DONE.

****

Clause 11.3: Add the following axiom: "$cos\_F(x) = 1$ if $|x| < 0.5 * \sqrt{epsilon}$"

DONE.

****

Clause 11.4: Replace the second axiom by $COS\_FF(0) = 1$.

DONE.

****

Clause 11.5: Replace $fmin_N$) by $r^{-1-p/2}$ in the first axiom.

NOT DONE; that axiom has been eliminated.

****

Clause 11.6: Insert $TAN\_FF(0, u) = 0$ as the first axiom.

DONE, but as the second axiom.

****

Clause 11.10: Replace the second axiom by $SEC\_FF(0, u) = 1$.

DONE.

****

Clause 12: If two-argument trig operations are eliminated, the two-argument inverse trig operations should also be eliminated.

Two argument operations have not been eliminated.

Otherwise, the minimum value for u should be the same as in clause 11.

DONE.

****

Clause 12.1: In the second axiom, replace $fmin_N$ by $r^{-p/2}$.

NOT DONE – the second axiom is replaced by "$arcsin(0) = 0$'.

****

Clause 12.2: This item has three parts:

Insert "underflow" in the signature between "**bad_argument**" and "}".

DONE.

Insert a new axiom: $ARCSIN\_FF(0, u) = 0$.

DONE.

Add the following "exception"

"Else if $|x| < (u * fmin_N)/(2 * \pi)$, $ARCSIN\_FF =$ **underflow**"

DONE.

****

Clause 12.4: Replace the third axiom by $ARCCOS\_FF(0, u) = u/4$.

DONE.

****

Clause 12.5: Replace $fmin_N$ in the second axiom by $r^{-p/2}$.

NOT DONE – the second axiom is replaced by "$arctan\_F(0) = 0$".

****

Clause 12.6: This item has three parts:

Insert "**underflow**" in the signature between "**bad_argument**" and "}".

DONE.

Insert the new axiom $ARCTAN\_FF(0, u) = 0$ between the first two axioms.

DONE.

Insert a second exception reading
"If $|x| < (2 * \pi/u) * fmin_N$, $ARCTAN\_FF(x, u) =$ **underflow**".

DONE.

****

Clause 12.7: Replace $fmin_N$ by $r^{-p/2}$ in the second exception.

NOT DONE – The second exception is correctly specified as it stands.

****

Clause 12.8: Replace the first line at the beginning of the Extensions component by:

"If $u \notin F$, $ARCTAN\_FFF(y, x, u) =$ **bad_argument**
Else, if $u \leq 0$, $ARCTAN\_FFF(y, x, u) =$ **bad_argument**
Else, in the table:"

DONE.

****

Clause 12.12: In the signature, insert the word "**underflow**" between "**bad_argument**" and "}".

The signature already contains "*udfl*".

****

Clause 12.16: This item has two parts:

In the signature insert the word "**underflow**" between "**bad_argument**" and "}".

DONE.

Add the exception

"Else if $|x| < (u * fmin_N)/(2 * \pi)$, $ARCCSC\_FF(x, u) =$ **underflow**."

DONE.

****

Clause 12.17: Add $arctan2\_360$ to the list.

The list is gone. Each trigonometric function with its argument in degrees is now listed with the corresponding functions with the argument in radians and arbitrary units u.

****

Clause 13.1: Replace the first axiom by "$SINH\_F(x) = x$ for $|x| < floor(r^{-p/2})$".

NO – it is replaced by "$sinh(0) = 0$" instead.

****

Clause 13.2: Replace the second axiom by "$COSH\_F(x) = 1$ for $|x| < floor(r^{-p/2})$"

DONE.

****

Clause 13.3: Replace the second axiom by the following three axioms:

"$TANH\_F(x) = x$ if $|x| < r{-}p/2$"

"$TANH\_F(x) = 1$ if $x > (p/2) * ln(r)$"

"$TANH\_F(x) = -1$ if $x < -(p/2) * ln(r)$"

DONE.

****

Clause 13.4: This item has three parts:

In the signature, add the word “**floating_overflow**” between “**pole**” and “}”.

DONE.

Insert the following two axioms

“$COTH\_F(x) = 1$ if $x \geq (p/2) * ln(r)$”

“$COTH\_F(x) = -1$ if $x \leq -(p/2) * ln(r)$”

DONE.

Add the following two exceptions:

“$coth\_F(x) =$ **floating_overflow**$(+\infty)$ if $0 < x < ceil(1/fmax$”

“$coth\_F(x) =$ **floating_overflow**$(-\infty)$ if $floor(-1/fmax) < x < 0$”

DONE.

****

Clause 13.5: Replace the second axiom by $SECH\_F(x) = 1$ if $|x| < floor(r^{-p})$

NO – it is replaced by “$sech(0) = 1$” instead.

****

Clause 13.6: Replace $1/fmax$ by $ln(2/fmax)$ in the second and third exceptions.

DONE.

Replace 0 by $-\mathbf{0}$ in third extension.

DONE.

****

Clause 14.1: This item has two parts:

Replace the current axioms by the following two axioms:

1. “$0 \leq ARCSINH\_F(x) \leq ln(2 * fmax)$”

   DONE.

2. “$ARCSINH\_F(x) = x$ if $|x| < r^{-p/2}$”

   NO – used instead: “$ARCSINH\_F(0) = 0$”.

****

Clause 14.2: This item has two parts:

In the first axiom, change “$1 \leq$” to “$0 \leq$”.

DONE.

In the exception replace $|x|$ by x.

DONE.

****

Clause 14.3: Replace the second axiom by "$ARCTANH\_F(x) = x$ if $|x| < r^{-p/2}$".

NO – replaced by "$arctan(0) = 0$" instead.

****

Clauses 15.1, 15.2, 15.3, and 15.4: In the signature delete the exception "**bad_argument**".

DONE IN ALL FOUR.

****

Clause 15.9: This item has two parts:

In the signature add the exception "**underflow**".

DONE.

Add a third exception:

"$CEILING\_F \to G =$ **underflow** if $|x| < (fmin_N)\_G$"

DONE.

Clause 15.10: This item has two parts:

In the signature add the exception "**underflow**".

DONE.

Add a third exception:

"$NEAREST\_F \to G =$ **underflow** if $|x| < (fmin_N)\_G$"

DONE.

****

Clause 15.11: This item has two parts:

In the signature add the exception "underflow".

DONE.

Add a third exception:

"$TRUNCATE\_F \to G =$ **underflow** if $|x| < (fmin_N)\_G$"

DONE.

****

Clause 15.12: This item has two parts:

In the signature add the exception "**underflow**".

DONE.

Add a third exception:

"$FLOOR\_F \to G =$ **underflow** if $|x| < (fmin_N)\_G$"

DONE.

****

NOTE: The following paragraph is now in clause 18:

The purpose of LIA-2's "doubled precision" operations is to expedite limited use of extended precision to maintain accuracy in (possibly) ill-conditioned applications. Such limited use often occurs in "inner loops" so that high performance is important. LIA-2 implicitly assumes that (in these applications) all input operands have been preprocessed so that any occurrence of a floating point exception indicates that the program is broken. See the paper "On the Orthogonality of Eigenvectors Computed by Divide-and-Conquer Techniques" by Sorenson and Tang in the SIAM Journal of Numerical Analysis, Vol. 28, No. 6 for a description of such limited use.

****

Clause 18.2.4: Make the following changes in Extensions:

Add the condition "if $|y| > 0$" to the first extension.

Then insert the following three additional extensions:

"$Rem\_DIV\_F(-0, 0) =$ **bad_argument**"

"$Rem\_DIV\_F(0, -0) =$ **bad_argument**"

"$Rem\_DIV\_F(-0, -0) =$ **bad_argument**"

DONE.

****

Clause 18.2.5: This item has three parts:

In the title replace the word "division" by "square root".

DONE.

In the signature, insert the word "**underflow**" between "{" and "**bad_argument**".

DONE.

Add the following exception:

"$REM\_SQRT\_F(x) =$ **underflow** if $x < (r^p) * fmin_N$"

DONE.

****

Clause 18.3: Add the following definition for a predicate for use in the doubled precision summation of p-digit quantities: "A doubled precision floating point quantity consists of two p-digit numbers: a high part, xhi, and a low part, xlo. If xlo is not zero, then its exponent is less than the exponent of xhi by at least $(p-1)$."

DONE.

****

Clause 18.3.1: Add "if xhi and xlo are the high and low parts of a doubled precision quantity" to the definition.

In the first exception, replace the "if ???" by "is permitted if $|xhi| > fmax/r^2$ and $|y| > fmax/r^2$".

In the second exception, replace the "if ???" by "is permitted if $|xhi| < (r^p) * fmin_N$ and $|y| < (r^p) * fmin_N$".

In the first extension, change the right hand side to $SUMMHI(0, xlo, y)$.

In the second extension, change the right hand side to $SUMMHI(xhi, 0, y)$.

In the third extension, change the right hand side to $SUMMHI(xhi, xlo, 0)$.

Remove the extensions dealing with input operands of infinity.

Append the following statement to the specification: "Overflow and underflow notifications are permitted when the conditions given above hold. In the absence of notifications the approximation constraints shall hold."

ALL OF THE ABOVE: DONE.

****

Clause 18.3.2: Add "if xhi and xlo are the high and low parts of a doubled precision quantity" to the definition.

In the first exception, replace the ??? by $SUMHI(xhi, xlo, y) =$ **floating_overflow**.

In the second exception, replace the ??? by $SUMHI(xhi, xlo, y) =$ **underflow**.

Add the following (third) exception: "It is permitted that

$SUMLO(xhi, xlo, y) =$ **underflow** if $|xhi + xlo + y - SUMHI(xhi, xlo, y)| < fmin_N$".

In the first extension, change the right hand side to $SUMMLO(0, xlo, y)$.

In the second extension, change the right hand side to $SUMMLO(xhi, 0, y)$.

In the third extension, change the right hand side to $SUMMLO(xhi, xlo, 0)$.

Remove the extensions dealing with input operands of infinity.

Append the following statement to the specification: "Overflow and underflow notifications are permitted when the conditions given above hold. In the absence of notifications the approximation constraints shall hold."

ALL OF THE ABOVE: DONE.

****

Clause 20.3: Add "**underflow**" to the signature.

DONE.

****

Clause A.5.1: Replace the last sentence by "For these operations the signature does not contain "**bad_argument**"."

DONE.

****

Clause A.5.3: In the last sentence, replace "may exist" by "are".

DONE.

****

Clause A.5.6.2: Replace "±infinity" by "$+\infty$ or $-\infty$".

DONE.

****

Clause A.5.6.4: In the last sentence put x in italics.

DONE.

****

Clause A.8: Remove the second paragraph.

DONE.

****

Clause A.8.1: Change the title to "Floating square root".

DONE.

In addition, the following first paragraph has been added:

> "There is no ambiguity in the result for $SQRT\_F(x)$: the existence of an ambiguity would require that the mathematical function could yield a result requiring exactly $(p + 1)$ digits for its representation. The square of such a number would require at least $(2p + 1)$ digits, which could not equal the p-digit number x."

****

Clause A.8.2: Insert the text: "The use of "nearest" to define SQRT_I produces no ambiguity in the result because the corresponding mathematical function cannot return a result exactly half way between two integers: the square of such a number could not be an integer."

DONE.

****

Clause A.8.3: Add the following sentence: "Such an operation involves division of each component of the vector by the magnitude of the vector or, equivalently and with higher performance, multiplication by the reciprocal of the magnitude."

DONE.

****

Clause A.9: In the second line replace x^y by $POWER\_FF(x, y)$.

In the third line replace the first 0^0 by $POWER\_FF(0, 0)$.

In the next to the last line replace 0^0 by $POWER\_FF(0, 0)$.

DONE.

****

Clause A.9.1: Replace "±infinity" by "$+\infty$ or $-\infty$".

DONE.

****

Clause A.9.2: In the title replace "(base e)" by "minus 1".

DONE.

****

Clause A.9.3: In the first line, remove ", x^y,".

In the second sentence of the third paragraph, put "k" in italics.

In the identity replace $POWER\_FF(x, y)$ by x^y.

ALL OF THE ABOVE: DONE.

****

Clause A.11: In the first indented line, replace "sin and" by "sec and".

Following "All four of" replace "these" by "the operations with poles".

ALL OF THE ABOVE: DONE.

****

Clause A.11.6: Use italics for $u/4$ and $u = 360$.

DONE.

****

Clause A.11.7: In the second line replace "values" by "value".

In the second line replace "±infinity" by "$+\infty$ or $-\infty$".

In the third line replace "±0" by "0 or −0".

ALL OF THE ABOVE: DONE.

****

Clause A.11.9: In the first line, put "sec(x)" in italics.

DONE.

****

Clause A.11.11: Replace "can be $(\pm infinity)$ for an argument of (±0)" by "is $+\infty$ or $-\infty$ for an argument of 0 or −0".

DONE.

****

Clause A.12: Replace the last sentence by "The undefined and underflow notifications are the only notifications produced by the inverse trigonometric operations."

DONE.

****

Clause A.12.7: Replace the word "saltus" by "discontinuity".

Insert the words "the corresponding mathematical function" after the words "limiting value of".

Replace $arctan2(y, x)$ by ", $arctan2(y, x)$,".

Replace "$\pm infinity$" by "$+\infty$ and $-\infty$".

In the line following the table, put b in italics.

ALL OF THE ABOVE: DONE.

****

Clause A.13.1: Insert ", or greater" at the end of the first sentence.

DONE.

****

Clause A.14.3: Replace "$x = \pm 1$" by "$x = +1$ and $x = -1$".
Clause A.14.4: Replace "$x = \pm 1$" by "$x = +1$ and $x = -1$".

DONE – both of them.

****

Clause A.15: In the second line replace "to both integer types and floating point types." by "to another integer type and to a floating point type."

DONE.

****

Clause A.16: Delete the period in "I/O."

DONE.

****

Clause 20.1: In the second sentence, replace "It can never" by "It must never".

DONE.

****

ANNEX B: In the second line replace "seleced" by "selected".

The introduction to Annex B has been reworded and expanded.

****

Annex C:

a) Change the title of the first section to "International Standards Documents".

b) Item 6: Replace the current text by a reference to Ada95.

c) Item 14: Replace the dash following "-1:" by "1994".

d) Change the title of the second section to "National Standards Documents".

e) Change the title of the third section to "Books, Articles, and Other Documents".

f) Item 25: Replace "W." by "W".

g) Item 26: Replace "W." by "W".

h) Item 26: Replace "microsystems" by "Microsystems".

ALL OF THE ABOVE: DONE.

Annex A: Response to Comments by Frank Farance

Summaries of the comments, together with the responses, follow.

****

Which ISO rules is this project operating under? The old rules or the new rules? This is important to determine since WG14 would need to gauge the amount of activity required for review.

> The LIA-2 project is working under the "old" rukes: therefore the next CD will not be designated as "final CD." However, commentors are requested to submit comments at their earliest convenience in order to avoid delays at the DIS level.

****

****

Subclause 2.1: Specifying "conformity with extensions" is not appropriate wording since the extensions defined in this subclause could be anything (including providing, say, I/O operations). Extensions should be limited to only those extensions specified by LIA-2. Additionally, the extensions in this subclause are unrelated to the extensions related to $-\mathbf{0}$, $+\infty$, $-\infty$.

> The use of the word "extensions" in this context has been eliminated.

****

Subclause 2.2: What happens if a binding standard does not provide all the operations? LIA-2 is missing a statement of conformance for a binding standard.

> It is intended that Items a) to h) in Clause 7 define conformance for a binding standard. Language committees are responsible for the production of binding standards.

****

Subclause 4.1: The phrases "real numbers" and "complex numbers" should have "mathematical" in front of them, just as there are "mathematical integers".

> Integers occur in many contexts, while real and complex numbers occur only in a mathematical context. Hence, it seems unnecessary to clutter them with the adjective "mathematical."

****

Subclause 4.1: Should be: "$\Rightarrow$ for logical implication, and $\Leftrightarrow$ for logical double implication".

> The editor sees no contradiction to the above statement in clause 4.1.

****

Subclause 4.1: The operator "$|$", used in the "GCD", "LCM", "EVEN", and "ODD" operators, is undefined. Additionally, this dyadic operator conflicts with its use as "such as" (see "GCD" and "LCM").

There are many mathematical symbols whose meaning depends on the context in which they appear. The editors consider it unlikely that anyone will be confused by the use of "|" for "such as" with its use to indicate "absolute value."

****

Subclause 4.1: More WG14 committee feedback is necessary on the "**pole**" and "**arg_too_big**" exceptional values. WG14 will investigate this over the next several meetings.

No action necessary until WG14 sends more information.

****

Subclause 4.1: Not all C implementations support $-\mathbf{0}$, **NaN**, $+\infty$, $-\infty$. In fact, some C implementations only support a single infinity.

LIA-2 states very clearly in clauses 5.6, 6, and 6.1 that specifications involving these symbols apply only to implementations which support them. It also requires an implementation which does not support them to document its alternatives.

****

Subclause 6.1: No mention is made of single infinities as continuation values. These should be considered since it is not clear how to map extensions of signed infinities into a single infinity.

The use of "single" (or "unsigned") infinities are already defined in several implementations. Inclusion of definitions in LIA-2 might introduce conflicts with existing practice.

****

Subclauses 18.3.1 and 18.3.2: The exceptions are missing conditions: "if ???".

LIA-2 now contains the appropriate conditions.

****

Subclauses 20.4 through 20.7: The use of the dyadic operator "|" is ambiguous. It means both the mathematical "such as" and some yet to be defined arithmetic operator.

See the third response to Subclause 4.1 above.

****

Annex A: There is no rationale for the extensions to the operators, i.e., the use of $-\mathbf{0}$, **NaN**, $+\infty$, and inf as operands.

The rationale for specifications dealing with these operands has been added to clause A.5.6.

****

Annex A.9: The term "X/OPEN" is not explained. A footnote should be added.

X/OPEN is a well known international organization; it needs no special comment. In Annex A.9, dealing with specifications for exponentiation, the sentence involving X/OPEN has been replaced by "The X/OPEN Portability Guide specifies a return value of 1."

****

Annex C: The bibliography should use ISO dates, e.g., 1993-06-29 / 1993-07-02 (a date range).

NOT DONE: Not worth the effort.

****

Annex C: The bibliography should include countries in addresses. For example: "AT&T Bell Laboratories, Murry Hill, N.J., USA".

NOT DONE: Not worth the effort.

****

Annex C: Books with ISBNs should be included. Magazines with ISSNs should be included.

NOT DONE: Not worth the effort.

****

Annex C: Item 9 refers to C90 (ISO/IEC 9899:1990) and C89 (ANSI X3.159-1989). They are not the same document. Either choose one, or include both as separate references.

Item 9 has been split into two items, dealing with the two references individually.

****

Annex C: Item 26 has a typo. It should be "Sun Microsystems".

The typo has been corrected.

****

Annex C: There should be consistent use of periods at the end of abbreviations (consult the ISO style guide). As an example, compare item 32 "Murry Hill, N.J." with item 33 "D C Sorenson and P T P Tang".

Such use of periods is now consistent; they are omitted in all abbreviations.

Annex B: Response to Comments by Fred Tydeman

Tydeman's comments start with the statement that LIA-2, as it now stands, should be rejected. Most of his comments list the errors, ambiguities, and other defects which he finds in the current version of LIA-2.

LIA-2 now includes the changes necessary to correct all errors and to include some of his suggestions for improvement.

****

Now that LIA-2 has introduced a "**pole**" exception, it seems that LIA-1 should be changed. Currently in LIA-1, $div(x, y)$ is undefined exception for any x when y is 0.0. This should be changed so that $div(0.0, 0.0)$ is **bad_argument** exception and $div(non - zero, 0.0)$ is **pole** exception. This makes LIA-1 more like LIA-2 and IEEE-754.

> There are many places in this document where closer alignment with the text in LIA-1 would be desirable. This problem will doubtless increase when LIA-3 is written. The editor's view is that each part should be as self-consistent as possible. At some time in the future it will be desirable to more closely align the three parts.

Consider adding "For x not an element F:" before each Extensions heading in sections 10 and higher. That would make it clearer that the exceptions section does not apply to $\pm infinity$ arguments.

> A statement to this effect is now in clause 5.

****

Page 6, 4.2 Definitions: Change "5.6" to "5.2" in 17) normalized.

> Done.

****

Page 8, 5.2 Definition. In requirement b) change the upper case OP to lower case op in the exponent of r so that the error is defined in terms of the mathematical result instead of the computed result.

> The error should be defined in terms of the computed result, because this is what is available to a program.

****

Page 10, 5.5 Error limit: Please add words here or in 4.2 Definitions on "rnd_error" which is used in many places elsewhere in the document, but appears not to be defined. I assume that this is 5.2.4; in LIA-1.

> The term "rounding error" has been added to the definitions in clause 4.2, and refers to clause 5.2.8 in LIA-1 for details.

****

Page 13, 8.1 $SQRT\_F$: Add "if $x < 0$" to exceptions, otherwise, $SQRT\_F$ is undefined for all arguments.

> DONE. All specifications (for all operations) now make clear the values of x to which they apply.

****

Page 13, 8.3 $RSQRT\_F$: Axioms: $RSQRT\_F(x) > 0$ should be $RSQRT\_F(x) \geq 0$.

> NO: There is no value of x in F for which $RSQRT\_F(x) = 0$.

****

Page 14, 9.1 EXP_F: Axioms: $EXP\_F(x) > 0$ should be $EXP\_F(x) \geq 0$.

> NO: There is no value of x in F for which $e^x = 0$.

****

Page 14, 9.2 $EXPM1\_F$: Signature: Add underflow.
Exceptions: Add $EXPM1f(x) =$ **underflow** if $|x| < fmin_N$.

> DONE.

****

Page 15, 9.3 $POWER\_FF$: This item has four parts:

> You need to allow for y being an odd integer in floating-point format) when $x < 0.0$ is valid (or overflow or underflow) instead of being undefined. Right now you have a conflict between the axiom $POWER\_FF(x, 1.0) = x$ and the exception $POWER\_FF(x, y) =$ **bad_argument** if $x < 0.0$ for the case $POWER\_FF(-1.0, 1.0)$.
>
> You should also allow for y being an even integer (in floating-point format) when $x < 0.0$ as being valid (or overflow or underflow) instead of being undefined.
>
> $POWER\_FF(0.0, y)$ for y an odd integer less than 0.0 should be a pole exception and have a suggested value of +infinity.
>
> $POWER\_FF(-0.0, y)$ for y an odd integer less than 0.0 should be a pole exception and have a suggested value of -infinity.
>
>> The arguments of $POWER\_FF$ are floating point numbers, and hence are approximate. LIA-2 does not give special treatment to such approximate integer values. Integer arguments are covered instead in $POWER\_FI$ and $POWER\_II$.
>
> Remove "if $x > 0$" from the axiom $POWER\_FF(x, 0) = 1$.
>
>> NO: This complements the exception for $x < 0$.
>
> Add $POWER\_FF(0.0, y) = pole(+/-INF)$ if y is a negative integer. The sign of the infinity depends upon the sign of 0.0 and if y is even or odd.
>
>> NO: See above on integer arguments for POWER_FF.

Remove the $POWER\_FF(-0, y) = POWER\_FF(0, y)$ statement from extensions as $1/(-0) \neq 1/(+0)$.

> DONE. Also the first two extensions (which are both wrong) have been changed to undefined.

****

Page 16 9.6.1 EXP_2_F: "$EXP\_2\_F > 0$", should be "$EXP\_2\_F(x) \geq 0$".

NO: There is no x in F producing equality to zero.

****

Page 17, 9.6.2 EXP_10_F: In Axioms: "$EXP\_10\_F(x) > 0$" should be "$EXP\_10\_F(x) \geq 0$".

NO: There is no x in F producing equality to zero.

****

On pages 17-19, the five logarithm functions in the extensions section have $LOG(+\infty) =$ **bad_argument**. I believe that they should be $LOG(+\infty) = +\infty$. In either case, rationale needs to be added to Annex A.10 explaining either choice.

NOT CHANGED: Logarithms of infinity should be undefined, partly because the true values should often be "in-range," – for example for an infinity produced by an overflow on an addition or subtraction. This justification is given in clause A.5.6.2.

****

Page 17, 10.1 LN_F: In the axiom change $fmin_N$ to $fmin$.

DONE.

****

Page 17, 10.2 $LN1P\_F$: This item has four parts:

Add underflow to the signature.

> DONE.

Change the first axiom to $LN1P\_F(x) = x$

> NO, this operations underflows for "$x < fmin_N$".

Add the axiom: $LN1P\_F(0) = 0$

> DONE.

Add the exception: $LN1P\_F(x) =$ **underflow** if $|x| < fmin_N$

> DONE.

****

Page 18, 8.3 LOG_FF: Add $b > 0$ and $b \neq 1$ to the axioms.

DONE.

****

Page 20, 11.1 SIN_F. This item has three parts.

Add axiom $SIN\_F(0) = 0$

NO: This is already implied by the second axiom.

Also remove "and $|x| > fmin_N$" from exceptions.

NO: This item covers underflow caused by argument reduction.

Also, change "$|sin(x)|$" to "$|SIN\_F(x)|$" in the exceptions. The problem I see is the mathematical function $sin(x)$ is $0.25 rnd\_error$ less than $fmin_N$, yet the computed value $SIN\_F(x)$ is one rnd_error greater than $fmin_N$, which is not an underflow. The other case is $sin(x)$, which is $0.25 rnd\_error$ greater than $fmin_N$, yet $SIN\_F(x)$ is one rnd_error less than $fmin_N$, which is an underflow. Both of these cases meet the required error limit. Underflow should be determined by the computed value, not the mathematical value.

The above is true, but the suggested use of SIN_F would produce a circular definition.

****

Page 20, 11.2 SIN_FF: This item has three parts:

Add the axiom $SIN\_FF(0, u) = 0$.

DONE.

Exceptions is missing the second "|" of absolute value symbol in the third line.

FIXED.

Remove "and $x > u/8$" from the exceptions.

NO: This covers underflow generated by argument reduction.

****

Page 21, 11.3 COS_F: This item has two parts:

Add the axiom "$COS\_F(0) = 1$".

DONE.

Change "$|cos(x)|$" to "$|COS\_F(x)|$" in the exceptions.

NO: This would make the definition of COS_F circular.

****

Page 22, 11.5 TAN_F: This item has four parts:

Add the axiom "$TAN\_F(0) = 0$".

DONE.

Remove ")" from "$fmin_N$)".

DONE.

Remove "and $|x| > 2 * fmin_N$" from the exceptions.

All of the exceptions are now more carefully written.

Change "$tan(x)$" to "$TAN\_F(x)$" three places in exceptions.

NO: This would make the definition of TAN_F circular.

****

25. Page 22, 11.6 TAN_FF. This item has three parts:

Add the axiom: $TAN\_FF(0, u) = 0$ if $u > 0$.

DONE.

Change the "==" to "=" two places in the exceptions.

DONE. Also the use of "mod" has been eliminated.

Remove "and $|x| > u/8$" in the exceptions.

NO: This covers underflow generated by argument reduction.

****

Pages 23-24 missing from my review copy.

****

Page 26, 12.1 $ARCSIN\_F$. This item has three parts:

Add underflow to signature.

DONE.

Add $ARCSIN\_F(0) = 0$ to axioms.

DONE.

Add "$ARCSIN\_F(x) =$ **underflow** if $|x| < fmin_N$" to exceptions.

DONE.

****

Page 26, 12.2 $ARCSIN\_FF$. This item has three parts:

Add underflow to signature.

DONE.

Add $ARCSIN\_FF(0, u) = 0$ if $u > 0$ to axioms.

DONE.

Add "$ARCSIN\_FF(x, u) =$ **underflow** if $|x * u/(2\pi)| < fmin_N$" to exceptions.

DONE; except that the correct condition is "if $x < u * fmin_N/(2\pi)$."

****

Page 27, 12.5 $ARCTAN\_F$. This item has three parts:

Add **underflow** to signature.

DONE.

Add "$ARCTAN\_F(0) = 0$" to axioms.

NO: This is implied by the second axiom.

Add "$ARCTAN\_F(x) =$ **underflow** if $|x| < fmin_N$" to exceptions.

DONE.

****

Page 28, 12.6 $ARCTAN\_FF$. This item has three parts:

Add underflow to signature.

DONE.

Add "$ARCTAN\_FF(0, u) = 0$ if $u > 0$" to axioms.

DONE.

Add "$ARCTAN\_FF(x) =$ **underflow** if $|x * u/(2pi)| < fmin_N$" to exceptions.

Done; except that the correct condition is "if $x < u * fmin_N/(2pi)$."

****

Page 29, 12.7 $ARCTAN2\_FF$. Flip the rows in the table so that

$y = +\infty$ is at the top and $y = -\infty$ is at the bottom.

Then the table would look like the Cartesian plane.

NO. The editor prefers the present ordering.

****

Page 29, 12.8 $ARCTAN2\_FFF$. This item has two parts:

Add "$u > 0$" to axioms.

IMPLIED: axioms have $u > umin$ with $umin$ a positive constant.

Add absolute value bars ($||$) to y/x in exceptions.

DONE.

****

Page 30, 12.9 $ARCCOT\_F$. This item has two parts:

Add "$ARCCOT\_F(0) = rnd(\pi/2)$" to axioms.

DONE; except "nearest" used instead of "rnd."

Add "$ARCCOT\_F(-0) = rnd(-\pi/2)$" to extensions.

DONE; except "nearest" used instead of "rnd."

****

Page 32, 12.16 $ARCCSC\_FF$. Change "$-u/4 < ARCCSC$" to "$-u/4 \leq ARCCSC$" in the axioms.

DONE.

****

Page 33, 13.1 $SINH\_F$: This item has three parts:

Add underflow to signature.

DONE.

Add "$SINH\_F(x) = x$ if $|x| < fmin_N$" to axioms.

NOT DONE – instead "$sinh(0) = 0$" is added.

Add "$SINH\_F(x) =$ **underflow** if $|x| < fmin_N$" to exceptions.

No. Underflow will not occur for x in F.

****

Page 33, 13.3 $TANH\_F$. This item has three parts:

Add underflow to signature.

DONE.

Add "$TANH\_F(x) = x$ if $|x| < fmin_N$" to axioms.

NOT DONE – instead "$tanh\_F(0) = 0$" is added.

Add "$TANH\_F(x) =$ **underflow** if $|x| < fmin_N$" to exceptions.

DONE.

****

Page 34, 13.5 $SECH\_F$. Change "$0 <$" to "$0 \leq$" in the axiom.

NO; no x in F yields "$sech(x) = 0$".

****

Page 35, 14.1 $ARCSINH\_F$. This item has three parts:

Add underflow to signature.

DONE.

Add "$ARCSINH\_F(x) = x$ if $|x| < fmin_N$" to axioms.

NOT DONE – instead "$ARCSINH\_F(0) = 0$" is added.

Add "$ARCSINH\_F(x) =$ **underflow** if $|x| < fmin_N$" to exceptions.

DONE.

****

Page 35, 14.2 $ARCCOSH\_F$. This item has two parts:

In first axiom, change "$1 \leq$" to "$0 \leq$".

In exceptions, change "$|x|$" to "x".

BOTH ITEMS DONE.

****

Page 35, 14.3 $ARCTANH\_F$. This item has three parts:

Add underflow to signature.

DONE.

Add "$ARCTANH\_F(x) = x$ if $|x| < fmin_N$" to axioms.

NOT DONE – instead "$ARCTANH\_F(0) = 0$" is added.

Add "$ARCTANH\_F(x) =$ **underflow** if $|x| < fmin_N$" to exceptions.

DONE.

****

Page 37, 15.1 $FLOOR\_F \rightarrow I$: Change the undefined to **integer_overflow** for the conversions from ±infinity.

NO; LIA-2 does not require support of **integer_overflow**.

****

Page 38, 15.2 $CEILING\_F \rightarrow I$: Change the undefined to **integer_overflow** for the conversions from ±infinity.

NO; LIA-2 does not require support of **integer_overflow**.

****

Page 38, 15.3 $NEAREST\_F \rightarrow I$: This item has four parts:

Change "$1/2 \leq x \leq maxint$" to "$1/2 < x < maxint$". $maxint$ may be odd, so, $maxint + 1/2$ round to nearest even is $maxint+1$, or overflow. $minint$ may be odd, so, $minint - 1/2$ round to nearest even is $minint - 1$ or overflow.

NO; The implementation should be free to choose how to handle "end-cases."

Change "$ifx > maxint$" to "if $x \geq maxint$" in exceptions.

NO; The implementation should be free to choose how to handle "end-cases."

Change "if $x < minint$" to "if $x \leq minint$" in exceptions.

NO; The implementation should be free to choose how to handle "end-cases."

Change the undefined to **integer_overflow** for the conversions from ± infinity.

NO; LIA-2 does not require support of **integer_overflow**.

****

Page 38, 15.4 $TRUNCATE\_F \rightarrow I$: Change the undefined to **integer_overflow** for the conversions from ±infinity.

NO; LIA-2 does not require support of **integer_overflow**.

****

Page 41, 16.1 External to internal. This item has two parts:

Add to the end of the Step (a) sentence: e.g., not a valid external form.

NO. The information it would supply is irrelevant.

Add "The same conversion function is used for all values, that is, $TRUNCATE\_IN\_F$ cannot be used for some values and $NEAREST\_IN\_F$ used for other values." to the paragraph that describes step (b).

NO. This statement is already implied.

****

Page 41, 16.1.1 $TRUNCATE\_IN\_F$. This item has two parts:

Change to domain from F to $\mathcal{R}$.

NO. The target domain is F.

The definition shows one argument, the extensions show three arguments. Fix the ones that are wrong.

DONE. The last two arguments have been removed.

****

Page 42, 16.1.2 $NEAREST\_IN\_F$. This item has two parts:

Change to domain from F to $\mathcal{R}$.

NO. The target domain is F.

The definition shows one argument, the extensions show three arguments. Fix the ones that are wrong.

DONE. The last two arguments have been removed.

****

Page 42, 16.2 Internal to external. Add "for all values" to the end of the paragraph that ends with $NEAREST\_ABS\_F$.

NO. This statement is already implied.

****

NOTE: The following paragraph is now in clause 18:

The comments on pages 47, 48, and 49 refer to "doubled precision" operations, whose purpose is rather limited. The purpose is to expedite limited use of extended precision to maintain accuracy in (possibly) ill-conditioned applications. Such limited use often occurs in "inner loops" so that high performance is important. LIA-2 implicitly assumes that (in these applications) all input operands have been preprocessed so that any occurrence of a floating point exception indicates that the program is broken. See the paper "On the Orthogonality of Eigenvectors Computed by Divide-and-Conquer Techniques" by Sorenson and Tang in the SIAM Journal of Numerical Analysis, Vol. 28, No. 6 for a description of such limited use.

****

Page 47, 18.2.1 $ADDLO\_F$: Change the undefined to 0 in the extensions in four places.

18.2.2 $SUBLO\_F$: Change the undefined to 0 in four places.

18.2.3 $MULLO\_F$: Change the undefined to 0 in four places.

NO TO ALL THREE OF THESE. LIA-2 does not provide the output for input of $+\infty$ or $-\infty$ for any "doubled precision" operation.

****

Page 48, 18.2.4 REM_DIV_F: This iten has two parts:

Add $REM\_DIV\_F(\pm INF, \pm INF)$ is undefined.

Change $REM\_DIV\_F(x, \pm INF)$ from undefined to x.

NO TO BOTH: LIA-2 does not provide the output for input of $+\infty$ or $-\infty$ for these operations.

****

Page 48, 18.2.5 REM_SQRT_F: This item has two parts:

"division" should be "sqrt" in the heading.

DONE.

Change $REM\_SQRT\_F(+INF)$ from undefined to 0.

NO. As noted above LIA-2 is not trying to supply complete multi-precision support.

****

Page 44, 18.3.1 $SUMHI\_F$: This item has four parts:

Change the first ??? to $|xhi + xlo + y| > fmax$.

DONE INSTEAD: The first "if ???" is replaced by "is permitted if $|xhi|$ and y both exceed $fmax/r^2$".

Change the second ??? to $|xhi + xlo + y| < fmin_N$.

DONE INSTEAD: The second "if ???" replaced by "is permitted if $|xhi|$ and y are both less than $r^p * fmin_N$".

Replace first undefined with "$xhi + xlo + y$ which is $+\infty$ or **bad_argument**"

Replace second undefined with "$xhi + xlo + y$ which is $-\infty$ or **bad_argument**"

NO TO BOTH OF THESE. The fourth and fifth extensions have been removed.

****

Page 49, 18.3.2 $SUMLO\_F$. This item has three parts:

Replace first ??? with

$|xhi + xlo + y - SUMHIf(xhi, xlo, y)| > fmax$.

NO. "???" has been replaced by "$SUMHI(xhi, xlo, y) =$ **floating_overflow**".

Replace second ??? with

$|xhi + xlo + y - SUMHI\_F(xhi, xlo, y)| < fmin_N$.

NO. "???" has been replaced by "$SUMHI(xhi, xlo, y) =$ **underflow**".

Replace the two undefined's with 0.

NO. LIA-2 does not provide the output for input of $+\infty$ or $-\infty$ for this operation..

****

Page 50, 19.1 $DPROD\_F \rightarrow G$: This item has two parts:

Add axiom $DPROD(x, y) = DPROD(y, x)$.

DONE.

Add extension: $DPROD(-\mathbf{0}, -\mathbf{0}) = 0$.

DONE.

****

Page 50, 19.2 MUL_ADD_F: This item has two parts:

Change the four undefined's to be "**bad_argument** or $+\infty$" or "**bad_argument** or $-\infty$" or "**bad_argument** or "±INF" as appropriate.

NO. The continuation values are left up to the implementation.

Add extension: $DPROD(-\mathbf{0}, -\mathbf{0}) = 0$.

DONE. Note: this is a repetition of the second item under DPROD.

****

Page 51, 20.1 $HYPOT\_F$: Add axiom $HYPOT\_F(x, y) < |x| + |y|$.

NO. This is too trivial.

****

Page 55, A.5.1 Signature. The second paragraph claims that undefined is in a signature even if it is only for operations with extended operands. That is false. For example, 19.1 $DPROD\_F- > G$ and 19.2 MUL_ADD_F both have undefined in extensions, yet neither has undefined in the signature.

True: The last sentence of this clause now reads "For these operations the signature does not contain **bad_argument**."

****

Page 55, A.5.2 Definition: This item has three parts:

Change the upper case OP to lower case op in the exponent of r in the definition of max_err_OP and in two places in the following paragraph.

NO; max_err_OP better enables a program to estimate the difference between the computed and true values.

Add that you are using ulps error in place of relative error and why.

DONE.

Add a reference to LIA-1, p.43, (V).

DONE.

****

Page 56, A.5.3 Axioms. Change $fmin$ to $fmin_N$ in the domain of EXP_F so that it matches 9.1.

DONE.

****

Page 57, Annex A.5.6 Extensions: There is no rationale for specific choices of operand ($-\mathbf{0}$, **NaN**, $+\infty$, and $-\infty$) and result in the extensions. For example, in subclause 17.2 (maximum sequence of floats), if the extensions are supported for $+\infty$, why isn't the result $+\infty$ for zero length arrays (as it is in APL)?. Some cases seem obvious, while other seem arbitrary. A simple 1-3 sentence rationale for each operator would be helpful.

NO. LIA-2 leaves it to the implementation to choose the output for input of $+\infty$ or $-\infty$.

****

Page 59, A.9.1 and A.9.2. Switch the words so that A.9.1 is EXP_F and A.9.2 is $EXPM1\_F$.

DONE.

****

Page 59, A.9.3 $POWER\_FF$. Add words about $POWER\_FF(negative, odd - integer)$.

DONE.

****

Page 61, A.10.2 $LNP1\_F$. Change that to $LN1P\_F$.

DONE.

****

Page 61, A.11 Trigonometric operations. "sin and tan have poles" should be "sec and tan have poles".

True: "*sin*" has been replaced by "*sec*."

****

Page 63, A.12 Inverse trigonometric operations. Add underflow to the list of notifications.

DONE.

****

Page 63, A.12.7 $ARCTAN2\_FF$. This item has three parts:

Change "function has a saltus" to "mathematical function has a discontinuity".

DONE.

Change $+\pi$ and $-\pi$ to $rnd(+\pi)$ and $rnd(-\pi)$.

DONE – except "nearest" is used instead of "rnd."

Change "the value 1 for" to "the value +0 for" for $ARCTAN2\_FF(0, 0)$.

NO. The phrase "the value 1" is quoted from a reference.

****

Page 64, A.12.8 $ARCTAN2\_FFF$. Change "function has a saltus" to "mathematical function has a discontinuity".

DONE.

****

Page 67, A.15.10 $CEILING\_F \rightarrow G$. "higherer" should be "higher".

DONE.

****

Page 67, A.16 External conversions. "I/O." should be "I/O".

DONE.

****

Page 70, A.18.2.4 REM_DIV_F. What is $DIVHI\_F(x, y)$?

See the last sentence of clause A.18.

Add the IEEE-754 remainder function (which is defined even if x/y would overflow or underflow) or explain why it was not in LIA-2.

NO. The explanation, provided in clause A.18, states that LIA-2 is not trying to provide complete support for multi-precision arithmetic.

****

Page 71, A.18.3 Doubled precision summation. Are there any requirements about the relationship between xhi and xlo? Such as $|xlo| < |xhi|$ or, even better, $|xlo| < |ulp(xhi)|$?

NO. However, the specifications imply the first, but not necessarily the second of the above inequalities. The second inequality implies a level of accuracy higher than is usually necessary for the intended applications.

****

Page 71, A.18.3.1 $SUMHI\_F$. What is the relationship between A.18.3.1 $SUMHI\_F$ and 18.3.1 $SUMHI\_F$? They have different exceptions and perhaps other differences.

A.18.3.2 $SUMLO\_F$. What is the relationship between A.18.3.2 $SUMLO\_F$ and 18.3.2 $SUMLO\_F$? They have different exceptions and perhaps other differences.

The conflicting (and erroneous) specifications have been removed.

****

Page 72, A.19.2 MUL_ADD_F. Must $MUL\_ADD(fmax, 2.0, -fmax)$ be $fmax$, even though the intermediate product $fmax * 2.0$ overflows? Must $MUL\_ADD(fmin, ulp(1.0), 1.0)$ suppress the intermediate underflow?

YES. This operation is intended to avoid the overhead of an exception, and thus improve performance.

****

Page 72, A.20.3 DIM_F. Explain why underflow depends upon denorm. In particular, if an implementation raises underflow when $|x - y| < fmin_N$, why is there no underflow here?

An add or sub operation should be used if it is desired to signal exceptions. In order to improve performance, DIM allows suppression of underflow by implementations which support denormalized numbers. However, on a system which does not support denormalized numbers, the occurrence of underflow must be permitted and signaled.

****

Page 74, Annex B. In the first paragraph, "seleced" should be "selected".

The introduction to Annex B has been expanded and reworded.

****

Page 76, Annex B. What is MOD_F? What is $POLY$?

They were left over from an earlier version; they have now been removed.

****

Page 77, Annex C Bibliography. [14] should be 10967-1:1994.

DONE.

****

Page 78, Annex C. [26] should "SUN microsystems" be "SUN Microsystems"?

YES – DONE.

****

Page 78, Annex C. [29] should be Chapter 5 Floating-Point C Extensions in Technical Report Numerical C Extensions Committee X3J11 April 1995 SC22/WG14 N403 X3J11/95-004.

DONE.