Proposal for C26
WG14 Nnnnn

Title:	**DRAFT** Annex G and F updates - Introduction
Author:	Damian McGuckin
Date:	2024-08-08
Proposal Category:	Editorial
Reference:	N3301

The changes listed here fall into two categories. Some make details of the various paragraphs more consistent. Others enhance readability or clarity, and in a very few cases, fill in gaps. Both Annex F and Annex G should now be consistent both internally and with each other especially in the use of commas, terms, sentence structure and ways of expressing the same sort of sentence in two different places,

Comments within this document including **ASIDE:** comments, and various editorial annotations used to highlight changes, will disappear from this document before submission.

Annex F consistency fixes

F.10.1 General - Paragraph 1 - REPLACE - For Consistency Between F+G

This clause contains specifications for `<math.h>` and `<tgmath.h>` functions that are particularly suited to ISO/IEC 60559 implementations.

F.10.1 General - Paragraph 2 - REPLACE - For Consistency Between F+G

The specifications in the following subclauses augment the definitions in `<math.h>` and `<tgmath.h>`. Where specifications appear for a function *f* which is a member of a family of functions, those specifications apply to all member functions of that family. even though only the (principal) member function is shown.

F.10.1 General - Paragraph 3 - REPLACE - For Consistency Between F+G

Unless otherwise specified, where the $\pm$ symbol occurs within both a function argument *f* and its result, it implies two cases: one with each $\pm$ symbol replaced with a plus (+) symbol in both argument and result, and the other with each $\pm$ symbol replaced with a minus (–) symbol in both argument and result.

F.10.1 General - Paragraph 14 - REWORD - For Consistency Between F+G

Functions with an argument of NaN return a result of NaN and raise no floating-point exception, except where explicitly stated otherwise.

F.10.1 General - Paragraph 14+ - was Paragraph 3 - Make English Consistent

For each function *f* of a single argument in `<math.h>` whose mathematical analog is an even function, then $f(-x) = f(x)$ for all the ISO/IEC 60559 rounding modes for all x in the domain of f. Likewise, for each function f of a single argument in `<math.h>` whose mathematical analog is an odd function, then $f(-x) = -f(x)$ for the roundTiesToEven, roundTiesToAway, and roundTowardZero ISO/IEC 60559 rounding modes for all x in the domain of f. The **`atan2`** and **`atan2pi`** functions are odd in their first ar gument.

ASIDE: Changed the concept of *mathematical counterpart* to *mathematical analog*. Why introduce the totally new term *counterpart* for a single use in this one place in the standard.

ASIDE: Individual functions are not identified as odd or even, unlike Annex G.

F.10.1 General - Paragraph 15 - REPLACE - was Paragraph 2

The macro **`HUGE_VAL`** and its **`float`** and **`long double`** analogs, **`HUGE_VALF`** and **`HUGE_VALL`**, expand to expressions whose values are positive infinities.

F.10.1 General - Paragraph 15+ - Add An Introduction like Annex G

The behavior of the special cases of the functions of Table F.3, including their treatment of any "invalid" and "divide-by-zero" floating-point exception, is specified in the following subclauses. As noted earlier, where the specifications for the principal function of a family of functions is seen, they apply to all member functions within that family.

Annex G consistency fixes

G.4.1 Paragraph 1 - For Consistency Between F+G

This clause contains specifications for `<complex.h>` functions that are particularly suited to ISO/IEC 60559 implementations.

G.4.1 Paragraph 1+ - For Consistency Between F+G

The specifications in the following subclauses augment the definitions in `<complex.h>`. Where specifications appear for a function *f* which is a member of a family of functions, those specifications apply to all member functions of that family. even though only the (principal) member function is shown.

G.4.1 Paragraph 1++ - For Consistency Between F+G - Multiple Paragraphs

Unless otherwise specified, where the $\pm$ symbol occurs within both a function argument *f* and its result, it implies two cases: one with each $\pm$ symbol replaced with a plus (+) symbol in both argument and result, and the other with each $\pm$ symbol replaced with a minus (–) symbol in both argument and result. Where the $\pm$ symbol occurs only within an argument and not in the function result, the result is independent of that sign. Where the $\pm$ symbol occurs only within a function result and not in an argument, the sign of that part of the result is unspecified.

A complex number *z* as used in the following subclauses can be written (6.2.5#17) as $z = x + i\,y$, i.e. as the sum of its real ($x = \mathbf{Re}(z)$) and imaginary ($y = \mathbf{Im}(z)$) components. Similarly, the complex conjugate of *z*, known also as $\overline{z}$ or conj(*z*), can be written as $\overline{z} = x - i\,y$ (7.3.9.4).

Each of the functions *f* seen in this annex generally takes a single complex number *z* as an argument. When *f* returns a complex number result, it is generally written as the sum of its real and imaginary components.

G.4.1. Paragraph 4 - Add this sentence

Where a NaN appears in a special case's argument(s) or result, even though its representation might have a sign bit, that bit has no interpretation.

G.4.1. Paragraph 4+ - For Consistency Between F+G

Functions with an argument of $\mathsf{NaN} + i\,\mathsf{NaN}$ return a result of $\mathsf{NaN} + i\,\mathsf{NaN}$ and raise no floating-point exception, except where explicitly stated otherwise.

DELETE the last item of G.4.2.1, G.6.3.1-6, G.6.4.1-2, and G.6.5.2 as this new paragraph now makes them redundant.

G.4.1 Paragraph 6 - table text now consistent with Paragraph 7

Each of the functions **`cabs`** and **`carg`** (7.3) is specified by a formula in terms of a real function (whose special cases were specified earlier in Annex F):

G.4.1 Paragraph 7 - table text now consistent with Paragraph 6

Each of the functions **`casin`**, **`catan`**, **`ccos`**, **`csin`**, and **`ctan`** (7.3) is specified by a formula in terms of some other complex function (whose special cases are specified later in this annex):

G.4.1 Paragraph 8 - Reword for clarity - Multiple Paragraphs

Each of **`cacos`**, **`cacosh`**, **`casinh`**, **`catanh`**, **`ccosh`**, **`csinh`**, **`ctanh`**, **`cexp`**, **`clog`**, and **`csqrt`** (7.3) is specified by a formula in terms of other real or complex functions. Each of these is an instance of a complex function f which can take an argument $z = x + i\,y$ in either the upper complex half-plane, i.e. $|x| < \infty$ with $0 < y < \infty$ or$y$ is a +0, or an ar gument $z = \overline{z} = x - i\,y$ in the lower complex half-plane. For each such function f, the conjugate of the evaluation of f for some ar gument z is identical to the evaluation of that same f for an argument of $\overline{z}$, the conjugate of that same z, or more compactly $\text{conj}\,(f(z)) = f(\text{conj}\,(z))$. This means that any specifications for the upper complex half-plane (i.e. where **Im***(z)* is always positively signed) imply specifications for the lower complex half-plane (i.e. where **Im***(z)* is always negatively signed). Additionally, if such a function f is either even or odd, i.e. it satisfies either $f(-z) = f(z)$ or $f(-z) = -f(z)$ respectively, then specifications for the first quadrant imply specifications for the other three quadrants.

The behavior of the special cases of these ten functions, including their treatment of any "invalid" and "divide-by-zero" floating-point exception, is specified in the following subclauses. As noted earlier, where the specifications for the principal function of a family of functions is seen, they apply to all member functions within that family.

ASIDE: Ensure the words about floating-point exceptions and the words about odd and even functions are consistent between Annex F and G. Note that, just as in Annex F, they hold beyond just special cases, even though those words here are only used in a special case context.