Draft DRs for TS 18661 **WG 14 N2029**

2016-03-19

==========

Reference Document: Floating Point Extensions, Part 1

Subject: Typos

Summary

1. Page 18: In C 7.6.1a#4, the last sentence, "functon" should be "function".

2. Page 48: In C 7.6.2.4a#3, "The **fetestexcept** function returns …" should be "The **fetestexceptflag** function returns …".

Suggested Technical Corrigendum

1. Page 18: In C 7.6.1a, paragraph 4, the last sentence, change "functon" to "function"

2. Page 48: In C 7.6.2.4a#3, change "**fetestexcept**" to "**fetestexceptflag**".

==========

Reference Document: Floating Point Extensions, Part 1

Subject: Functions that round result to narrower type don't always

Summary

Page 38: The C 7.12.13a subclause heading is "Functions that round result to narrower type" and this is the way the functions in the subclause are referred to throughout the TS. In some cases, the functions in the subclause round their result to a type that isn't really narrower than the parameter types. For example, this is true for the functions **daddl**, **dsubl**, etc. if the **long double** and **double** types have the same width (as is allowed). (With the extended types introduced in TS 18661-3, the destination type might be wider, as it might for **f32xaddf64**.)

The current way of referencing these functions reflects the usual situation, and is perhaps a helpful way of think about them generally. With a note about the uncharacteristic cases, it seems unlike to cause significant confusion. Also, changing all the references to these functions would be a large editorial undertaking, spanning multiple parts of the TS. Confusion could easily arise from having an inconsistent set of documents.

Suggested Technical Corrigendum

Page 38: After the C 7.12.13a subclause heading, insert the following paragraph:

> [1] The functions in this subclause round their results to a type typically narrower than the parameter types.

Page 40: After the change to C ending with "7.12.13a.6 Square root rounded to narrower type … [3] These functions return the square root of x, rounded to the type of the function.", insert the following:

> In 7.12.13a #1, attach a footnote to the wording:
>
> > typically narrower
>
> where the footnote is:
>
> > *) In some cases the destination type might not be narrower than the parameter types. For example, **double** might not be narrower than **long double**.

==========

Reference Document: Floating Point Extensions, Part 1

Subject: Specification for nonexistent case

Summary

Page 51: C 7.25#6a says "The functions that round result to a narrower type have type-generic macros whose names are obtained by omitting any **f** or **l** suffix from the function names", but there are no such functions with an **f** suffix.

Though not strictly incorrect, the current specification might cause confusion by leading a user to look for the nonexistent functions. If the mention of **f** suffixes is omitted, a footnote might be helpful in explaining why **f** suffixes are not relevant.

Suggested Technical Corrigendum

Page 51: In C 7.25#6a, change "by omitting any **f** or **l** suffix" to "by omitting any **l** suffix".

Page 51: After the change to C that ends with 7.25#6b, insert:

> In 7.25#6a, attach a footnote to the wording:
>
> > **l** suffix

where the footnote is:

*) There are no functions with these macro names and the **f** suffix.

==========

Reference Document: Floating Point Extensions, Part 1

Subject: feature macros and header file inclusions

Summary

ISO/IEC TS 18661-1 subclause 5.3 specifies interfaces that are defined or declared "only if **__STDC_WANT_IEC_60559_BFP_EXT__** is defined as a macro at the point in the source file where the header for the interface is first included." C 7.12#1 says **<tgmath.h>** includes **<math.h>** and **<complex.h>**.

So for

```
#include <math.h>
#define __STDC_WANT_IEC_60559_BFP_EXT__
#include <tgmath.h>
float f(float x) { return nextup(x); }
```

the **nextup** functions in **<math.h>** are not declared and the **nextup** macro in **<tgmath.h>** is defined. Since **x** has type **float**, the function determined by the **nextup** macro in **<tgmath.h>** is **nextupf**. But is this function available to be called?

Another example. For

```
#include <limits.h>
#define __STDC_WANT_IEC_60559_BFP_EXT__
#include <math.h>
…
```

the **fromfp** functions in **<math.h>** are declared, but the **WIDTH** macros in **<limits.h>**, which are needed for portable use of the **fromfp** functions, are not defined.

In these examples, interfaces provided by one header are related to interfaces that are not provided by another header, because of the placement of the **WANT** macros. This leads to ambiguous cases (as in the first example above) and incomplete feature sets. Later parts of the TS have their own **WANT** macros, which compounds the problem. See also Joseph Myers's <http://www.open-std.org/jtc1/sc22/wg14/13831>.

The suggested corrigendum below specifies that the same set of **WANT** macros must be defined at the points in the code where the relevant headers are first included. This results in fewer combinations of interfaces and provides one sets of interfaces that is consistent

and complete with respect to a given set of **WANT** macros.

Suggested Technical Corrigendum

Page 5: At the end of 5.3, insert:

After 7.1.2#4, insert:

> [4a] Some standard headers define or declare identifiers contingent on whether certain macros whose names begin with **_STDC_WANT_IEC_60559_** and end with **_EXT_** are defined (by the user) at the point in the code where the header is first included. Within a preprocessing translation unit, the same set of such macros shall be defined for the first inclusion of all such headers.

==========

Reference Document: Floating Point Extensions, Part 2

Subject: Typos

Summary

1. Page viii: In the Purpose, second paragraph, a "/" is missing in "ISO/IECIEEE 60559:2011", which should be "ISO/IEC/IEEE 60559:2011".

2. P 34: In 12.3, the C 7.12.13a title is written as "Functions that round result to narrower format", but the title should be "Functions that round result to narrower type", as given in ISO/IEC TS 18661-1.

Suggested Technical Corrigendum

1. Page viii: In the Purpose, second paragraph, change "ISO/IECIEEE 60559:2011" to "ISO/IEC/IEEE 60559:2011".

2. P 34: In 12.3, change the C 7.12.13a title from "Functions that round result to narrower format" to "Functions that round result to narrower type".

==========

Reference Document: Floating Point Extensions, Part 3

Subject: Typos

Summary

1. Page 39: In 12.3, the line

    7.12.14 Functions that round result to narrower format

contains two editorial errors. First, the intended subclause (introduced in ISO/IEC TS 18661-1) is 7.12.13a, not 7.12.14. Second, the title of the subclause is "Functions that round result to narrower type".

Suggested Technical Corrigendum

Page 39: In 12.3, change the line

    7.12.14 Functions that round result to narrower format

to

    7.12.13a Functions that round result to narrower type

==========

Reference Document: Floating Point Extensions, Part 3

Subject: Error in function name

Summary

Page 32: In 12.3, the function name is written as "**scoshdNx**", instead of "**coshdNx**" as intended.

Although correcting the mistake could be seen as a substantive change, it is clear from the context that this function is in the family of **cosh** functions. It is extremely unlikely that any implementer would not have recognized the mistake and provided the function with the erroneous name.

Suggested Technical Corrigendum

Page 32: In 12.3, change "**scoshdNx**" to "**coshdNx**".

==========