Proposal for C2X
WG14 N2469

Title:	NaN and infinity macros
Author, affiliation:	C FP group
Date:	2020-01-03
Proposal category:	Defect
Reference:	N2454 C2X working draft

Problem:

In C2X 7.12, NaN and infinity macros (**`SNAN`** with an optional type suffix, **`NAN`**, **`INFINITY`**, and the corresponding decimal macros) are defined to expand to constant expressions. 6.6#8 says

> An *arithmetic constant expression* shall have arithmetic type and shall only have operands that are integer constants, floating constants, enumeration constants, character constants, **`sizeof`** expressions whose results are integer constants, and **`_Alignof`** expressions. …

Floating constants just represent numbers, not NaNs or infinities. An expression with only floating constant operands (or the other permitted operands) can't evaluate to a signaling NaN. Expressions with floating constant operands can evaluate to a quiet NaN or an infinity, but not without an "invalid" or "divide-by-zero" floating-point exception. These macros (at least for IEC 60559 implementations) are not intended to raise floating-point exceptions and need to be implemented with compiler intrinsics.

The **`HUGE_VAL`** macros have the same problem. They are defined to be constant expressions and footnote 242 says they "can be positive infinities in an implementation that supports infinities."

Thus, the NaN, infinity, and **`HUGE_VAL`** macros are defined in way that makes them unimplementable. The following suggested changes (1) loosen the definitions so that the macros expand to expressions instead of constant expressions and (2) say that the macros may be used wherever a constant expression of the appropriate type may be used.

Suggested change:

In 7.12#5, change:

The macro **`HUGE_VAL`** expands to a positive **`double`** constant expression, not necessarily representable as a **`float`**. …

to:

The macro **`HUGE_VAL`** expands to a positive **`double`** expression, not necessarily representable as a **`float`**. It may be used wherever a constant expression of type **`double`** may be used. …

In 7.12#6, change:

The macro **`HUGE_VAL_D32`** expands to a constant expression of type **`_Decimal32`** representing positive infinity. …

to:

The macro **`HUGE_VAL_D32`** expands to an expression of type **`_Decimal32`** representing positive infinity. It may be used wherever a constant expression of type **`_Decimal32`** may be used. …

In 7.12#7, change:

The macro **`INFINITY`** expands to a constant expression of type **`float`** representing positive or unsigned infinity, if available; else to a positive constant of type **`float`** that overflows at translation time.

to:

The macro **`INFINITY`** expands to an expression of type **`float`** representing positive or unsigned infinity, if available; else to a positive constant of type **`float`** that overflows at translation time. It may be used wherever a constant expression of type **`float`** may be used.

In 7.12#8, change:

The macro **`DEC_INFINITY`** expands to a constant expression of type **`_Decimal32`** representing positive infinity.

to:

The macro **`DEC_INFINITY`** expands to an expression of type **`_Decimal32`** representing positive infinity. It may be used wherever a constant expression of type **`_Decimal32`** may be used.

In 7.12#9, change:

The macro **`NAN`** …. It expands to a constant expression of type **`float`** representing a quiet NaN.

to:

The macro **`NAN`** …. It expands to an expression of type **`float`** representing a quiet NaN. It may be used wherever a constant expression of type **`float`** may be used.

In 7.12#10, change:

The macro **`DEC_NAN`** expands to a constant expression of type **`_Decimal32`** representing a quiet NaN.

to:

The macro **`DEC_NAN`** expands to an expression of type **`_Decimal32`** representing a quiet NaN. It may be used wherever a constant expression of type **`_Decimal32`** may be used.

In 7.12#11, change:

The signaling NaN macros **`SNANF`** … They expand to a constant expression of the respective type representing a signaling NaN.

to:

The signaling NaN macros **`SNANF`** … They expand to an expression of the respective type representing a signaling NaN. They may be used wherever a constant expression of the respective type may be used.

In 7.12#12, change:

The decimal signaling NaN macros **`SNAND32`** … each expands to a constant expression of the respective decimal floating type representing a signaling NaN.

to:

The decimal signaling NaN macros **`SNAND32`** … each expands to an expression of the respective decimal floating type representing a signaling NaN. They may be used wherever a constant expression of the respective type may be used.