

ieee_arithmetic: IEEE Arithmetic Facilities Module

July 25, 2022

1 Name

`ieee_arithmetic` — Intrinsic module providing IEEE arithmetic facilities

2 Usage

USE,INTRINSIC :: IEEE_ARITHMETIC

This module provides various facilities related to IEEE arithmetic.

The contents of this module conform to technical report ISO/IEC TR 15580:1998(E). Some additional features from the Fortran 2018 standard (ISO/IEC 1539-1:2018) are included.

3 Synopsis

Derived Types

`IEEE_CLASS_TYPE`, `IEEE_FLAG_TYPE` (from `IEEE_EXCEPTIONS`), `IEEE_ROUND_TYPE`, `IEEE_STATUS_TYPE` (from `IEEE_EXCEPTIONS`).

Parameters

`IEEE_ALL` (from `IEEE_EXCEPTIONS`), `IEEE_DIVIDE_BY_ZERO` (from `IEEE_EXCEPTIONS`), `IEEE_DOWN`, `IEEE_INEXACT` (from `IEEE_EXCEPTIONS`), `IEEE_INVALID` (from `IEEE_EXCEPTIONS`), `IEEE_NEAREST`, `IEEE_NEGATIVE_DENORMAL`, `IEEE_NEGATIVE_SUBNORMAL*`, `IEEE_NEGATIVE_INF`, `IEEE_NEGATIVE_NORMAL`, `IEEE_NEGATIVE_ZERO`, `IEEE_OTHER`, `IEEE_OVERFLOW` (from `IEEE_EXCEPTIONS`), `IEEE_POSITIVE_DENORMAL`, `IEEE_POSITIVE_SUBNORMAL*`, `IEEE_POSITIVE_INF`, `IEEE_POSITIVE_NORMAL`, `IEEE_POSITIVE_ZERO`, `IEEE_QUIET_NAN`, `IEEE_SIGNALING_NAN`, `IEEE_TO_ZERO`, `IEEE_UNDERFLOW` (from `IEEE_EXCEPTIONS`), `IEEE_UP`, `IEEE_USUAL` (from `IEEE_EXCEPTIONS`).

* These are from the Fortran 2018 standard.

Operators

`==`, `/=`.

Procedures

`IEEE_CLASS`, `IEEE_COPY_SIGN`, `IEEE_GET_FLAG` (from `IEEE_EXCEPTIONS`), `IEEE_GET_HALTING_MODE` (from `IEEE_EXCEPTIONS`), `IEEE_GET_ROUNDING_MODE`, `IEEE_GET_STATUS` (from `IEEE_EXCEPTIONS`), `IEEE_IS_FINITE`, `IEEE_IS_NAN`, `IEEE_IS_NEGATIVE`, `IEEE_IS_NORMAL`, `IEEE_LOGB`, `IEEE_NEXT_AFTER`, `IEEE_NEXT_DOWN*`, `IEEE_NEXT_UP*`, `IEEE_REM`, `IEEE_RINT`, `IEEE_SCALB`, `IEEE_SELECTED_REAL_KIND`, `IEEE_SET_FLAG` (from `IEEE_EXCEPTIONS`), `IEEE_SET_HALTING_MODE` (from `IEEE_EXCEPTIONS`), `IEEE_SET_ROUNDING_MODE`, `IEEE_SET_STATUS` (from `IEEE_EXCEPTIONS`), `IEEE_SUPPORT_DATATYPE`, `IEEE_SUPPORT_DENORMAL`, `IEEE_SUPPORT_DIVIDE`, `IEEE_SUPPORT_FLAG` (from `IEEE_EXCEPTIONS`), `IEEE_SUPPORT_HALTING` (from `IEEE_EXCEPTIONS`), `IEEE_SUPPORT_INF`, `IEEE_SUPPORT_NAN`, `IEEE_SUPPORT_ROUNDING`,

IEEE_SUPPORT_SQRT, IEEE_SUPPORT_SUBNORMAL* IEEE_SUPPORT_STANDARD,
IEEE_UNORDERED, IEEE_VALUE.

* These are from the Fortran 2018 standard.

4 Derived-Type Description

```
TYPE IEEE_CLASS_TYPE
  PRIVATE
  ...
END TYPE
```

Type for specifying the class of a number. Its only possible values are those of the named constants exported by this module.

```
USE, INTRINSIC :: IEEE_EXCEPTIONS, ONLY: IEEE_FLAG_TYPE
```

See IEEE_EXCEPTIONS for a description of this type.

```
TYPE IEEE_ROUND_TYPE
  PRIVATE
  ...
END TYPE
```

Type for specifying the rounding mode. Its only possible values are those of the named constants exported by this module.

```
USE, INTRINSIC :: IEEE_EXCEPTIONS, ONLY: IEEE_STATUS_TYPE
```

See IEEE_EXCEPTIONS for a description of this type.

5 Parameter Description

```
USE, INTRINSIC :: IEEE_EXCEPTIONS, ONLY: IEEE_ALL
```

See IEEE_EXCEPTIONS for a description of this parameter.

```
USE, INTRINSIC :: IEEE_EXCEPTIONS, ONLY: IEEE_DIVIDE_BY_ZERO
```

See IEEE_EXCEPTIONS for a description of this parameter.

```
TYPE(IEEE_ROUND_TYPE), PARAMETER :: IEEE_DOWN
```

The rounding mode in which the results of a calculation are rounded to the nearest machine-representable number that is less than the true result.

USE,INTRINSIC :: IEEE_EXCEPTIONS,ONLY:IEEE_INEXACT

See IEEE_EXCEPTIONS for a description of this parameter.

USE,INTRINSIC :: IEEE_EXCEPTIONS,ONLY:IEEE_INVALID

See IEEE_EXCEPTIONS for a description of this parameter.

TYPE(IEEE_ROUND_TYPE),PARAMETER :: IEEE_NEAREST

The rounding mode in which the results of a calculation are rounded to the nearest machine-representable number.

TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_NEGATIVE_DENORMAL

A negative number whose precision is less than that of the normal numbers; the result of an IEEE gradual underflow.

TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_NEGATIVE_INF

Negative infinity.

TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_NEGATIVE_NORMAL

A normal negative number.

TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_NEGATIVE_SUBNORMAL

A negative number whose precision is less than that of the normal numbers; this constant is from Fortran 2018, and has the same value as IEEE_NEGATIVE_DENORMAL.

TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_NEGATIVE_ZERO

Negative zero.

TYPE(IEEE_ROUND_TYPE),PARAMETER :: IEEE_OTHER

Any processor-dependent rounding mode other than IEEE_DOWN, IEEE_NEAREST, IEEE_TO_ZERO and IEEE_UP.

USE,INTRINSIC :: IEEE_EXCEPTIONS,ONLY:IEEE_OVERFLOW

See IEEE_EXCEPTIONS for a description of this parameter.

TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_POSITIVE_DENORMAL

A positive number whose precision is less than that of the normal numbers; the result of an IEEE gradual underflow.

```
TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_POSITIVE_INF
```

Positive infinity.

```
TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_POSITIVE_NORMAL
```

A normal positive number.

```
TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_POSITIVE_SUBNORMAL
```

A positive number whose precision is less than that of the normal numbers; this constant is from Fortran 2018, and has the same value as `IEEE_NEGATIVE_DENORMAL`.

```
TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_POSITIVE_ZERO
```

Positive zero.

```
TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_QUIET_NAN
```

A “Not-a-Number” value that propagates through arithmetic operations but which does not necessarily raise the `IEEE_INVALID` exception on use.

```
TYPE(IEEE_CLASS_TYPE),PARAMETER :: IEEE_SIGNALING_NAN
```

A “Not-a-Number” that raises the `IEEE_INVALID` exception on use.

```
TYPE(IEEE_ROUND_TYPE),PARAMETER :: IEEE_TO_ZERO
```

The rounding mode in which the results of a calculation are rounded to the nearest machine-representable number that lies between zero and the true result.

```
USE,INTRINSIC :: IEEE_EXCEPTIONS,ONLY:IEEE_UNDERFLOW
```

See `IEEE_EXCEPTIONS` for a description of this parameter.

```
TYPE(IEEE_ROUND_TYPE),PARAMETER :: IEEE_UP
```

The rounding mode in which the results of a calculation are rounded to the nearest machine-representable number that is greater than the true result.

```
USE,INTRINSIC :: IEEE_EXCEPTIONS,ONLY:IEEE_USUAL
```

See `IEEE_EXCEPTIONS` for a description of this parameter.

6 Operator Description

In addition to ISO/IEC TR 15580:1998(E), the module IEEE_ARITHMETIC defines the ‘==’ and ‘/=’ operators for the IEEE_CLASS_TYPE. These may be used to test the return value of the IEEE_CLASS function. E.g

```
USE,INTRINSIC :: IEEE_ARITHMETIC, ONLY: IEEE_CLASS, &
  IEEE_QUIET_NAN, OPERATOR(==)
...
IF (IEEE_CLASS(X)==IEEE_QUIET_NAN) THEN
...

```

7 Procedure Description

```
ELEMENTAL TYPE(IEEE_CLASS_TYPE) FUNCTION IEEE_CLASS(X)
REAL(any kind),INTENT(IN) :: X

```

Returns the IEEE class that the value of X falls into.

```
ELEMENTAL REAL(kind) FUNCTION IEEE_COPY_SIGN(X,Y)
REAL(kind),INTENT(IN) :: X
REAL(kind),INTENT(IN) :: Y

```

Returns the value of X with the sign of Y. The result has the same kind as X.

This function is only available if IEEE_SUPPORT_DATATYPE(X) and IEEE_SUPPORT_DATATYPE(Y) are both true.

```
USE,INTRINSIC :: IEEE_EXCEPTIONS,ONLY:IEEE_GET_FLAG

```

See IEEE_EXCEPTIONS for a description of this procedure.

```
USE,INTRINSIC :: IEEE_EXCEPTIONS,ONLY:IEEE_GET_HALTING_MODE

```

See IEEE_EXCEPTIONS for a description of this procedure.

```
SUBROUTINE IEEE_GET_ROUNDING_MODE(ROUND_VALUE)
TYPE(IEEE_ROUND_TYPE),INTENT(OUT) :: ROUND_VALUE

```

Sets ROUND_VALUE to the current rounding mode, one of IEEE_DOWN, IEEE_NEAREST, IEEE_OTHER, IEEE_TO_ZERO or IEEE_UP.

```
SUBROUTINE IEEE_GET_UNDERFLOW_MODE(GRADUAL)
LOGICAL,INTENT(OUT) :: GRADUAL

```

This procedure was added in Fortran 2003. It sets GRADUAL to whether the current underflow mode permits gradual underflow (subnormal results), rather than abrupt underflow (subnormal results flushed to zero). It may only be invoked if IEEE_SUPPORT_UNDERFLOW_CONTROL(X) is .TRUE. for some kind of X. From Fortran 2018, GRADUAL may be any kind of LOGICAL.

```
USE,INTRINSIC :: IEEE_EXCEPTIONS,ONLY:IEEE_GET_STATUS
```

See IEEE_EXCEPTIONS for a description of this procedure.

```
ELEMENTAL LOGICAL FUNCTION IEEE_IS_FINITE(X)
REAL(any kind),INTENT(IN) :: X
```

Returns true if X is a finite number, i.e. neither an infinity nor a NaN.

```
ELEMENTAL LOGICAL FUNCTION IEEE_IS_NAN(X)
REAL(any kind),INTENT(IN) :: X
```

Returns true if X is a NaN (either quiet or signalling).

```
ELEMENTAL LOGICAL FUNCTION IEEE_IS_NEGATIVE(X)
REAL(any kind),INTENT(IN) :: X
```

Returns true if X is negative, even for negative zero.

```
ELEMENTAL LOGICAL FUNCTION IEEE_IS_NORMAL(X)
REAL(any kind),INTENT(IN) :: X
```

Returns if X is normal, i.e. not an infinity, a NaN, or denormal.

```
ELEMENTAL REAL(kind) FUNCTION IEEE_LOGB(X)
REAL(kind),INTENT(IN) :: X
```

Returns the unbiased exponent of X. For normal, non-zero numbers this is the same as the EXPONENT(X)-1; for zero, IEEE_DIVIDE_BY_ZERO is signalled and the result is negative infinity (or -HUGE(X) if negative infinity is not available); for an infinity the result is positive infinity; for a NaN the result is a quiet NaN.

```
ELEMENTAL REAL(kind) FUNCTION IEEE_NEXT_AFTER(X,Y)
REAL(kind),INTENT(IN) :: X
REAL(kind),INTENT(IN) :: Y
```

Returns the closest machine-representable number to X (of the same kind as X) that is either greater than X (if X<Y) or less than X (if X>Y). If X and Y are equal, X is returned. If the result is subnormal, IEEE_UNDERFLOW is signalled. If the result is infinite but X is finite, IEEE_OVERFLOW is signalled.

```
ELEMENTAL REAL(kind) FUNCTION IEEE_NEXT_DOWN(X)
REAL(kind),INTENT(IN) :: X
```

Returns the closest machine-representable number to X (of the same kind as X) that is less than X (unless X is $-\infty$ or a NaN; if X is a signalling NaN the result is a quiet NaN, otherwise X is returned). No exception is signalled unless X is a signalling NaN. This function is from Fortran 2018.

```
ELEMENTAL REAL(kind) FUNCTION IEEE_NEXT_UP(X)
REAL(kind),INTENT(IN) :: X
```

Returns the closest machine-representable number to X (of the same kind as X) that is greater than X (unless X is $+\infty$ or a NaN; if X is a signalling NaN the result is a quiet NaN, otherwise X is returned). No exception is signalled unless X is a signalling NaN. This function is from Fortran 2018.

```
ELEMENTAL REAL(kind) FUNCTION IEEE_REM(X,Y)
REAL(kind),INTENT(IN) :: X
REAL(kind),INTENT(IN) :: Y
```

The result value is the exact remainder from the division X/Y , viz $X-Y*N$ where N is the nearest integer to the true result of X/Y .

```
ELEMENTAL REAL(kind) FUNCTION IEEE_RINT(X)
REAL(kind),INTENT(IN) :: X
```

X rounded to the nearest integer using the current rounding mode.

```
ELEMENTAL REAL(kind) FUNCTION IEEE_SCALB(X,I)
REAL(kind),INTENT(IN) :: X
INTEGER(any kind),INTENT(IN) :: I
```

The result is $X*2**I$ without computing $2**I$, with overflow or underflow exceptions signalled only if the end result overflows or underflows.

```
INTEGER FUNCTION IEEE_SELECTED_REAL_KIND(P,R,RADIX)
INTEGER(any kind),INTENT(IN),OPTIONAL :: P
INTEGER(any kind),INTENT(IN),OPTIONAL :: R
INTEGER(any kind),INTENT(IN),OPTIONAL :: RADIX
```

The same as the intrinsic function `SELECTED_REAL_KIND(P,R,RADIX)`, but only returns numbers of kinds for which `IEEE_SUPPORT_DATATYPE` returns true.

```
USE,INTRINSIC :: IEEE_EXCEPTIONS,ONLY:IEEE_SET_FLAG
```

See `IEEE_EXCEPTIONS` for a description of this procedure.

```
USE,INTRINSIC :: IEEE_EXCEPTIONS,ONLY:IEEE_SET_HALTING_MODE
```

See `IEEE_EXCEPTIONS` for a description of this procedure.

```
SUBROUTINE IEEE_SET_ROUNDING_MODE(ROUND_VALUE)
TYPE(IEEE_ROUND_TYPE),INTENT(IN) :: ROUND_VALUE
```

Sets the current rounding mode to `ROUND_VALUE`. This is only allowed when `IEEE_SUPPORT_ROUNDING(ROUND_VALUE,X)` is true for all X such that `IEEE_SUPPORT_DATATYPE(X)` is true.

```
SUBROUTINE IEEE_SET_UNDERFLOW_MODE(GRADUAL)
LOGICAL,INTENT(IN) :: GRADUAL
```

This procedure was added in Fortran 2003. It sets the current underflow mode to “gradual underflow” (subnormal values may be produced) if `GRADUAL` is `.TRUE.`, and to “abrupt underflow” (subnormal results are flushed to zero) otherwise. It may only be invoked if `IEEE_SUPPORT_UNDERFLOW_CONTROL(X)` is `.TRUE.` for some kind of `X`. From Fortran 2018, `GRADUAL` may be any kind of `LOGICAL`.

```
USE, INTRINSIC :: IEEE_EXCEPTIONS, ONLY: IEEE_SET_STATUS
```

See `IEEE_EXCEPTIONS` for a description of this procedure.

```
LOGICAL FUNCTION IEEE_SUPPORT_DATATYPE(X)  
REAL(any kind), INTENT(IN), OPTIONAL :: X
```

Returns true if and only if all reals (if `X` is absent), or reals of the same kind as `X` conform to the IEEE standard for representation, addition, subtraction and multiplication when the operands and results have normal values.

```
LOGICAL FUNCTION IEEE_SUPPORT_DENORMAL(X)  
REAL(any kind), INTENT(IN), OPTIONAL :: X
```

Returns true if and only if IEEE denormalised values are supported for all real kinds (if `X` is absent) or for reals of the same kind as `X`.

```
LOGICAL FUNCTION IEEE_SUPPORT_DIVIDE(X)  
REAL(any kind), INTENT(IN), OPTIONAL :: X
```

Returns true if and only if division on all reals (if `X` is absent) or on reals of the same kind as `X` is performed to the accuracy required by the IEEE standard.

```
USE, INTRINSIC :: IEEE_EXCEPTIONS, ONLY: IEEE_SUPPORT_FLAG
```

See `IEEE_EXCEPTIONS` for a description of this procedure.

```
USE, INTRINSIC :: IEEE_EXCEPTIONS, ONLY: IEEE_SUPPORT_HALTING
```

See `IEEE_EXCEPTIONS` for a description of this procedure.

```
LOGICAL FUNCTION IEEE_SUPPORT_INF(X)  
REAL(any kind), INTENT(IN), OPTIONAL :: X
```

Returns true if and only if IEEE infinities are supported for all reals (if `X` is absent) or for reals of the same kind as `X`.

```
LOGICAL FUNCTION IEEE_SUPPORT_NAN(X)  
REAL(any kind), INTENT(IN), OPTIONAL :: X
```

Returns true if and only if IEEE NaNs are supported for all reals (if `X` is absent) or for reals of the same kind as `X`.


```

LOGICAL FUNCTION IEEE_SUPPORT_ROUNDING(ROUND_VALUE,X)
TYPE(IEEE_ROUND_TYPE) :: ROUND_VALUE
REAL(any kind),OPTIONAL :: X

```

Returns true if and only if the rounding mode for all reals (if X is absent) or reals of the same kind as X, can be changed to the specified rounding mode by the IEEE_SET_ROUNDING procedure.

```

LOGICAL FUNCTION IEEE_SUPPORT_SQRT(X)
REAL(any kind),INTENT(IN),OPTIONAL :: X

```

Returns true if and only if the SQRT intrinsic conforms to the IEEE standard for all reals (if X is absent) or for reals of the same kind as X.

```

LOGICAL FUNCTION IEEE_SUPPORT_SUBNORMAL(X)
REAL(any kind),INTENT(IN),OPTIONAL :: X

```

Returns true if and only if IEEE subnormal values are supported for all real kinds (if X is absent) or for reals of the same kind as X. This function is from Fortran 2018.

```

LOGICAL FUNCTION IEEE_SUPPORT_STANDARD(X)
REAL(any kind),INTENT(IN),OPTIONAL :: X

```

Returns true if and only if all the other IEEE_SUPPORT inquiry functions return the value true for all reals (if X is absent) or for reals of the same kind as X.

```

LOGICAL FUNCTION IEEE_SUPPORT_UNDERFLOW_CONTROL(X)
REAL(any kind),INTENT(IN),OPTIONAL :: X

```

This enquiry function was added in Fortran 2003. It returns true if and only if control of the underflow mode (gradual or abrupt) is available for all reals (if X is absent) or for reals of the same kind as X.

```

ELEMENTAL LOGICAL FUNCTION IEEE_UNORDERED(X,Y)
REAL(any kind),INTENT(IN) :: X
REAL(any kind),INTENT(IN) :: Y

```

Returns IEEE_IS_NAN(X).OR.IEEE_IS_NAN(Y).

```

ELEMENTAL REAL(kind) FUNCTION IEEE_VALUE(X,CLASS)
REAL(kind),INTENT(IN) :: X
TYPE(IEEE_CLASS_TYPE),INTENT(IN) :: CLASS

```

Returns a sample value of the same kind as X that falls into the specified IEEE number class. For a given kind of X and class, the same value is always returned.

8 See Also

nagfor(1), **ieee_exceptions**(3), **ieee_features**(3), **intro**(3), **nag_modules**(3).

9 Bugs

Please report any bugs found to ‘support@nag.co.uk’ or ‘support@nag.com’, along with any suggestions for improvements.