RenderScript Mathematical Constants and Functions

Overview

The mathematical functions below can be applied to scalars and vectors. When applied to vectors, the returned value is a vector of the function applied to each entry of the input.

For example:

```
float3 a, b;
// The following call sets
// a.x to sin(b.x),
// a.y to sin(b.y), and
// a.z to sin(b.z).
a = sin(b);
```

See Vector Math Functions for functions like distance() and length() that interpret instead the input as a single vector in n-dimensional space.

The precision of the mathematical operations on 32 bit floats is affected by the pragmas rs_fp_relaxed and rs_fp_full. Under rs_fp_relaxed, subnormal values may be flushed to zero and rounding may be done towards zero. In comparison, rs_fp_full requires correct handling of subnormal values, i.e. smaller than 1.17549435e-38f. rs_fp_rull also requires round to nearest with ties to even.

Different precision/speed tradeoffs can be achieved by using variants of the common math functions. Functions with a name starting with

- native_: May have custom hardware implementations with weaker precision. Additionally, subnormal values may be flushed to zero, rounding towards zero may be used, and NaN and infinity input may not be handled correctly.
- half_: May perform internal computations using 16 bit floats. Additionally, subnormal values may be flushed to zero, and rounding towards zero may be used.

Summary

Constants	
M_1_PI	1 / pi, as a 32 bit float
M_2_PI	2 / pi, as a 32 bit float
M_2_SQRTPI	2 / sqrt(pi), as a 32 bit float
M_E	e, as a 32 bit float
M_LN10	log_e(10), as a 32 bit float
M_LN2	log_e(2), as a 32 bit float
M_LOG10E	log_10(e), as a 32 bit float
M_LOG2E	log_2(e), as a 32 bit float
M_PI	pi, as a 32 bit float
M_PI_2	pi / 2, as a 32 bit float
M_PI_4	pi / 4, as a 32 bit float
M_SQRT1_2	1 / sqrt(2), as a 32 bit float
M_SQRT2	sqrt(2), as a 32 bit float

Functions	
abs	Absolute value of an integer
acos	Inverse cosine
acosh	Inverse hyperbolic cosine
acospi	Inverse cosine divided by pi
asin	Inverse sine
asinh	Inverse hyperbolic sine
asinpi	Inverse sine divided by pi
atan	Inverse tangent
atan2	Inverse tangent of a ratio
atan2pi	Inverse tangent of a ratio, divided by pi
atanh	Inverse hyperbolic tangent
atanpi	Inverse tangent divided by pi
cbrt	Cube root
ceil	Smallest integer not less than a value
clamp	Restrain a value to a range
clz	Number of leading 0 bits
copysign	Copies the sign of a number to another
cos	Cosine
cosh	Hypebolic cosine
cospi	Cosine of a number multiplied by pi
degrees	Converts radians into degrees
erf	Mathematical error function
erfc	Mathematical complementary error function
exp	e raised to a number
exp10	10 raised to a number
exp2	2 raised to a number
expm1	e raised to a number minus one
fabs	Absolute value of a float
fdim	Positive difference between two values
floor	Smallest integer not greater than a value
fma	Multiply and add
fmax	Maximum of two floats
fmin	Minimum of two floats
fmod	Modulo
fract	Positive fractional part
frexp	Binary mantissa and exponent
half_recip	Reciprocal computed to 16 bit precision
half_rsqrt	Reciprocal of a square root computed to 16 bit precision
half_sqrt	Square root computed to 16 bit precision

hypot	Hypotenuse
ilogb	Base two exponent
ldexp	Creates a floating point from mantissa and exponent
lgamma	Natural logarithm of the gamma function
log	Natural logarithm
log10	Base 10 logarithm
log1p	Natural logarithm of a value plus 1
log2	Base 2 logarithm
logb	Base two exponent
mad	Multiply and add
max	Maximum
min	Minimum
mix	Mixes two values
modf	Integral and fractional components
nan	Not a Number
nan_half	Not a Number
native_acos	Approximate inverse cosine
native_acosh	Approximate inverse hyperbolic cosine
native_acospi	Approximate inverse cosine divided by pi
native_asin	Approximate inverse sine
native_asinh	Approximate inverse hyperbolic sine
native_asinpi	Approximate inverse sine divided by pi
native_atan	Approximate inverse tangent
native_atan2	Approximate inverse tangent of a ratio
native_atan2pi	Approximate inverse tangent of a ratio, divided by pi
native_atanh	Approximate inverse hyperbolic tangent
native_atanpi	Approximate inverse tangent divided by pi
native_cbrt	Approximate cube root
native_cos	Approximate cosine
native_cosh	Approximate hypebolic cosine
native_cospi	Approximate cosine of a number multiplied by pi
native_divide	Approximate division
native_exp	Approximate e raised to a number
native_exp10	Approximate 10 raised to a number
native_exp2	Approximate 2 raised to a number
native_expm1	Approximate e raised to a number minus one
native_hypot	Approximate hypotenuse
native_log	Approximate natural logarithm
native_log10	Approximate base 10 logarithm

native_log1p	Approximate natural logarithm of a value plus 1
native_log2	Approximate base 2 logarithm
native_powr	Approximate positive base raised to an exponent
native_recip	Approximate reciprocal
native_rootn	Approximate nth root
native_rsqrt	Approximate reciprocal of a square root
native_sin	Approximate sine
native_sincos	Approximate sine and cosine
native_sinh	Approximate hyperbolic sine
native_sinpi	Approximate sine of a number multiplied by pi
native_sqrt	Approximate square root
native_tan	Approximate tangent
native_tanh	Approximate hyperbolic tangent
native_tanpi	Approximate tangent of a number multiplied by pi
nextafter	Next floating point number
pow	Base raised to an exponent
pown	Base raised to an integer exponent
powr	Positive base raised to an exponent
radians	Converts degrees into radians
remainder	Remainder of a division
remquo	Remainder and quotient of a division
rint	Round to even
rootn	Nth root
round	Round away from zero
rsRand	Pseudo-random number
rsqrt	Reciprocal of a square root
sign	Sign of a value
sin	Sine
sincos	Sine and cosine
sinh	Hyperbolic sine
sinpi	Sine of a number multiplied by pi
sqrt	Square root
step	0 if less than a value, 0 otherwise
tan	Tangent
tanh	Hyperbolic tangent
tanpi	Tangent of a number multiplied by pi
tgamma	Gamma function
trunc	Truncates a floating point

Deprecated Functions

rsClamp	**Deprecated**. Restrain a value to a range
rsFrac	**Deprecated**. Returns the fractional part of a float

Constants

M_1_PI : 1 / pi, as a 32 bit float

Value: 0.318309886183790671537767526745028724f

The inverse of pi, as a 32 bit float.

M_2_PI : 2 / pi, as a 32 bit float

Value: 0.63661977236758134307553505349005744 8f

2 divided by pi, as a 32 bit float.

M_2_SQRTPI : 2 / sqrt(pi), as a 32 bit float

Value: 1.12837916709551257389615890312154517 2f

2 divided by the square root of pi, as a 32 bit float.

M_E : e, as a 32 bit float

Value: 2.71828182845904523536028747135266249 8f

The number e, the base of the natural logarithm, as a 32 bit float.

M_LN10 : log_e(10), as a 32 bit float

Value: 2.30258509299404568401799145468436420 8f

The natural logarithm of 10, as a 32 bit float.

M_LN2 : log_e(2), as a 32 bit float

Value: 0.69314718055994530941723212145817656 8f

The natural logarithm of 2, as a 32 bit float.

M_LOG10E : log_10(e), as a 32 bit float

Value: 0.43429448190325182765112891891660508 2f

The logarithm base 10 of e, as a 32 bit float.

M_LOG2E : log_2(e), as a 32 bit float

Value: 1.44269504088896340735992468100189213 7f

The logarithm base 2 of e, as a 32 bit float.

M_PI : pi, as a 32 bit float

Value: 3.14159265358979323846264338327950288 4f

The constant pi, as a 32 bit float.

M_PI_2 : pi / 2, as a 32 bit float

Value: 1.57079632679489661923132169163975144 2f

Pi divided by 2, as a 32 bit float.

M_PI_4 : pi / 4, as a 32 bit float

Value: 0.78539816339744830961566084581987572 1f

Pi divided by 4, as a 32 bit float.

M_SQRT1_2 : 1 / sqrt(2), as a 32 bit float

Value: 0.70710678118654752440084436210484903 9f

The inverse of the square root of 2, as a 32 bit float.

M_SQRT2 : sqrt(2), as a 32 bit float

Value: 1.41421356237309504880168872420969807 9f

The square root of 2, as a 32 bit float.

Functions

abs : Absolute value of an integer

uchar abs(char v);
uchar2 abs(char2 v);
uchar3 abs(char3 v);
uchar4 abs(char4 v);
uint abs(int v);
uint2 abs(int2 v);
uint3 abs(int3 v);
uint4 abs(int4 v);
ushort abs(short v);
ushort2 abs(short2 v);
ushort3 abs(short3 v);
ushort4 abs(short4 v);

Returns the absolute value of an integer.

For floats, use fabs().

acos : Inverse cosine

float acos(float v);

float2 acos(float2 v);

float3 acos(float3 v);

float4 acos(float4 v);

half acos(half v); Added in API level 24

half2 acos(half2 v); Added in API level 24

half3 acos(half3 v); Added in API level 24

half4 acos(half4 v); Added in API level 24

Returns the inverse cosine, in radians.

See also native_acos().

acosh : Inverse hyperbolic cosine

float acosh(float v);

float2 acosh(float2 v);

float3 acosh(float3 v);

float4 acosh(float4 v);

half acosh(half v); Added in API level 24

half2 acosh(half2 v); Added in API level 24

half3 acosh(half3 v); Added in API level 24

half4 acosh(half4 v); Added in API level 24

Returns the inverse hyperbolic cosine, in radians.

See also native_acosh().

acospi : Inverse cosine divided by pi

float acospi(float v);

float2 acospi(float2 v);

float3 acospi(float3 v);

float4 acospi(float4 v);

half acospi(half v); Added in API level 24

half2 acospi(half2 v); Added in API level 24

half3 acospi(half3 v); Added in API level 24

half4 acospi(half4 v); Added in API level 24

Returns the inverse cosine in radians, divided by pi.

To get an inverse cosine measured in degrees, use `acospi(a) * 180.f`.

See also native_acospi().

asin : Inverse sine

float asin(float v);

float2 asin(float2 v);

float3 asin(float3 v);

float4 asin(float4 v);

half asin(half v); Added in API level 24

half2 asin(half2 v); Added in API level 24

half3 asin(half3 v); Added in API level 24

half4 asin(half4 v); Added in API level 24

Returns the inverse sine, in radians.

See also native_asin().

asinh : Inverse hyperbolic sine

float asinh(float v);

float2 asinh(float2 v);

float3 asinh(float3 v);

float4 asinh(float4 v);

half asinh(half v); Added in API level 24

half2 asinh(half2 v); Added in API level 24

half3 asinh(half3 v); Added in API level 24

half4 asinh(half4 v); Added in API level 24

Returns the inverse hyperbolic sine, in radians.

See also native_asinh().

asinpi : Inverse sine divided by pi

float asinpi(float v);

float2 asinpi(float2 v);

float3 asinpi(float3 v);

float4 asinpi(float4 v);

half asinpi(half v); Added in API level 24

half2 asinpi(half2 v); Added in API level 24

half3 asinpi(half3 v); Added in API level 24

half4 asinpi(half4 v); Added in API level 24

Returns the inverse sine in radians, divided by pi.

To get an inverse sine measured in degrees, use `asinpi(a) * 180.f`.

See also native_asinpi().

atan : Inverse tangent

float atan(float v);

float2 atan(float2 v);

float3 atan(float3 v);

float4 atan(float4 v);

half atan(half v); Added in API level 24

half2 atan(half2 v); Added in API level 24

half3 atan(half3 v); Added in API level 24

half4 atan(half4 v); Added in API level 24

Returns the inverse tangent, in radians.

See also native_atan().

atan2 : Inverse tangent of a ratio

float atan2(float numerator, float denominator);

float2 atan2(float2 numerator, float2 denominator);

float3 atan2(float3 numerator, float3 denominator);

float4 atan2(float4 numerator, float4 denominator);

half atan2(half numerator, half denominator); Added in API level 24

half2 atan2(half2 numerator, half2 denominator); Added in API level 24

half3 atan2(half3 numerator, half3 denominator); Added in API level 24

half4 atan2(half4 numerator, half4 denominator); Added in API level 24

Parameters

numerator Numerator.

denominator Denominator. Can be 0.

Returns the inverse tangent of `(numerator / denominator)`, in radians.

See also native_atan2().

atan2pi : Inverse tangent of a ratio, divided by pi

float atan2pi(float numerator, float denominator);

float2 atan2pi(float2 numerator, float2 denominator);

float3 atan2pi(float3 numerator, float3 denominator);

float4 atan2pi(float4 numerator, float4 denominator);

half atan2pi(half numerator, half denominator); Added in API level 24

half2 atan2pi(half2 numerator, half2 denominator); Added in API level 24

half3 atan2pi(half3 numerator, half3 denominator); Added in API level 24

half4 atan2pi(half4 numerator, half4 denominator); Added in API level 24

Parameters

numerator Numerator.

denominator Denominator. Can be 0.

Returns the inverse tangent of `(numerator / denominator)`, in radians, divided by pi.

To get an inverse tangent measured in degrees, use `atan2pi(n, d) * 180.f`.

See also native_atan2pi().

atanh : Inverse hyperbolic tangent

float atanh(float v);

float2 atanh(float2 v);

float3 atanh(float3 v);

float4 atanh(float4 v);

half atanh(half v); Added in API level 24

half2 atanh(half2 v); Added in API level 24

half3 atanh(half3 v); Added in API level 24

half4 atanh(half4 v); Added in API level 24

Returns the inverse hyperbolic tangent, in radians.

See also native_atanh().

atanpi : Inverse tangent divided by pi

float atanpi(float v);

float2 atanpi(float2 v);
float3 atanpi(float3 v);

float4 atanpi(float4 v);

half atanpi(half v); Added in API level 24

half2 atanpi(half2 v); Added in API level 24

half3 atanpi(half3 v); Added in API level 24

half4 atanpi(half4 v); Added in API level 24

Returns the inverse tangent in radians, divided by pi.

To get an inverse tangent measured in degrees, use `atanpi(a) * 180.f`.

See also native_atanpi().

cbrt : Cube root

float cbrt(float v);

float2 cbrt(float2 v);

float3 cbrt(float3 v);

float4 cbrt(float4 v);

half cbrt(half v); Added in API level 24

half2 cbrt(half2 v); Added in API level 24

half3 cbrt(half3 v); Added in API level 24

half4 cbrt(half4 v); Added in API level 24

Returns the cube root.

See also native_cbrt().

ceil : Smallest integer not less than a value

float ceil(float v);

float2 ceil(float2 v);

float3 ceil(float3 v);

float4 ceil(float4 v);

half ceil(half v); Added in API level 24

half2 ceil(half2 v); Added in API level 24

half3 ceil(half3 v); Added in API level 24

half4 ceil(half4 v); Added in API level 24

Returns the smallest integer not less than a value.

For example, `ceil(1.2f)` returns 2.f, and `ceil(-1.2f)` returns -1.f.

See also floor().

clamp : Restrain a value to a range

char clamp(char value, char min_value, char max_value); Added in API level 19

char2 clamp(char2 value, char min_value, char max_value); Added in API level 19

char2 clamp(char2 value, char2 min_value, char2 max_value); Added in API level 19

char3 clamp(char3 value, char min_value, char max_value); Added in API level 19

char3 clamp(char3 value, char3 min_value, char3 max_value); Added in API level 19

char4 clamp(char4 value, char min_value, char max_value); Added in API level 19

char4 clamp(char4 value, char4 min_value, char4 max_value); Added in API level 19

float clamp(float value, float min_value, float max_value);	
float2 clamp(float2 value, float min_value, float max_value);	
float2 clamp(float2 value, float2 min_value, float2 max_value);	
float3 clamp(float3 value, float min_value, float max_value);	
float3 clamp(float3 value, float3 min_value, float3 max_value);	
float4 clamp(float4 value, float min_value, float max_value);	
float4 clamp(float4 value, float4 min_value, float4 max_value);	
half clamp(half value, half min_value, half max_value);	Added in API level 24
half2 clamp(half2 value, half min_value, half max_value);	Added in API level 24
half2 clamp(half2 value, half2 min_value, half2 max_value);	Added in API level 24
half3 clamp(half3 value, half min_value, half max_value);	Added in API level 24
half3 clamp(half3 value, half3 min_value, half3 max_value);	Added in API level 24
half4 clamp(half4 value, half min_value, half max_value);	Added in API level 24
half4 clamp(half4 value, half4 min_value, half4 max_value);	Added in API level 24
int clamp(int value, int min_value, int max_value);	Added in API level 19
int2 clamp(int2 value, int min_value, int max_value);	Added in API level 19
int2 clamp(int2 value, int2 min_value, int2 max_value);	Added in API level 19
int3 clamp(int3 value, int min_value, int max_value);	Added in API level 19
int3 clamp(int3 value, int3 min_value, int3 max_value);	Added in API level 19
int4 clamp(int4 value, int min_value, int max_value);	Added in API level 19
int4 clamp(int4 value, int4 min_value, int4 max_value);	Added in API level 19
long clamp(long value, long min_value, long max_value);	Added in API level 19
long2 clamp(long2 value, long min_value, long max_value);	Added in API level 19
long2 clamp(long2 value, long2 min_value, long2 max_value);	Added in API level 19
long3 clamp(long3 value, long min_value, long max_value);	Added in API level 19
long3 clamp(long3 value, long3 min_value, long3 max_value);	Added in API level 19
long4 clamp(long4 value, long min_value, long max_value);	Added in API level 19
long4 clamp(long4 value, long4 min_value, long4 max_value);	Added in API level 19
short clamp(short value, short min_value, short max_value);	Added in API level 19
short2 clamp(short2 value, short min_value, short max_value);	Added in API level 19
short2 clamp(short2 value, short2 min_value, short2 max_value);	Added in API level 19
short3 clamp(short3 value, short min_value, short max_value);	Added in API level 19
short3 clamp(short3 value, short3 min_value, short3 max_value);	Added in API level 19
short4 clamp(short4 value, short min_value, short max_value);	Added in API level 19
short4 clamp(short4 value, short4 min_value, short4 max_value);	Added in API level 19
uchar clamp(uchar value, uchar min_value, uchar max_value);	Added in API level 19
uchar2 clamp(uchar2 value, uchar min_value, uchar max_value);	Added in API level 19
uchar2 clamp(uchar2 value, uchar2 min_value, uchar2 max_value);	Added in API level 19
uchar3 clamp(uchar3 value, uchar min_value, uchar max_value);	Added in API level 19
uchar3 clamp(uchar3 value, uchar3 min_value, uchar3 max_value);	Added in API level 19
uchar4 clamp(uchar4 value, uchar min_value, uchar max_value);	Added in API level 19
uchar4 clamp(uchar4 value, uchar4 min_value, uchar4 max_value);	Added in API level 19
uint clamp(uint value, uint min_value, uint max_value);	Added in API level 19
uint2 clamp(uint2 value, uint min_value, uint max_value);	Added in API level 19
uint2 clamp(uint2 value, uint2 min_value, uint2 max_value);	Added in API level 19
uint3 clamp(uint3 value, uint min_value, uint max_value);	Added in API level 19
uint3 clamp(uint3 value, uint3 min_value, uint3 max_value);	Added in API level 19

uint4 clamp(uint4 value, uint min_value, uint max_value);	Added in API level 19
uint4 clamp(uint4 value, uint4 min_value, uint4 max_value);	Added in API level 19
ulong clamp(ulong value, ulong min_value, ulong max_value);	Added in API level 19
ulong2 clamp(ulong2 value, ulong min_value, ulong max_value);	Added in API level 19
ulong2 clamp(ulong2 value, ulong2 min_value, ulong2 max_value);	Added in API level 19
ulong3 clamp(ulong3 value, ulong min_value, ulong max_value);	Added in API level 19
ulong3 clamp(ulong3 value, ulong3 min_value, ulong3 max_value);	Added in API level 19
ulong4 clamp(ulong4 value, ulong min_value, ulong max_value);	Added in API level 19
ulong4 clamp(ulong4 value, ulong4 min_value, ulong4 max_value);	Added in API level 19
ushort clamp(ushort value, ushort min_value, ushort max_value);	Added in API level 19
ushort2 clamp(ushort2 value, ushort min_value, ushort max_value);	Added in API level 19
ushort2 clamp(ushort2 value, ushort2 min_value, ushort2 max_value);	Added in API level 19
ushort3 clamp(ushort3 value, ushort min_value, ushort max_value);	Added in API level 19
ushort3 clamp(ushort3 value, ushort3 min_value, ushort3 max_value);	Added in API level 19
ushort4 clamp(ushort4 value, ushort min_value, ushort max_value);	Added in API level 19
ushort4 clamp(ushort4 value, ushort4 min_value, ushort4 max_value);	Added in API level 19

Parameters

value	Value to be clamped.
min_value	Lower bound, a scalar or matching vector.
max_value	High bound, must match the type of low.

Clamps a value to a specified high and low bound. clamp() returns min_value if value < min_value, max_value if value > max_value, otherwise value.

There are two variants of clamp: one where the min and max are scalars applied to all entries of the value, the other where the min and max are also vectors.

If min_value is greater than max_value, the results are undefined.

clz : Number of leading 0 bits

char clz(char value);
char2 clz(char2 value);
char3 clz(char3 value);
char4 clz(char4 value);
int clz(int value);
int2 clz(int2 value);
int3 clz(int3 value);
int4 clz(int4 value);
short clz(short value);
short2 clz(short2 value);
short3 clz(short3 value);
short4 clz(short4 value);
uchar clz(uchar value);
uchar2 clz(uchar2 value);
uchar3 clz(uchar3 value);
uchar4 clz(uchar4 value);
uint clz(uint value);
uint2 clz(uint2 value);

uint3 clz(uint3 value);
uint4 clz(uint4 value);

ushort clz(ushort value);

ushort2 clz(ushort2 value);

ushort3 clz(ushort3 value);

ushort4 clz(ushort4 value);

Returns the number of leading 0-bits in a value.

For example, `clz((char)0x03)` returns 6.

copysign : Copies the sign of a number to another

float copysign(float magnitude_value, float sign_value);

float2 copysign(float2 magnitude_value, float2 sign_value);

float3 copysign(float3 magnitude_value, float3 sign_value);

float4 copysign(float4 magnitude_value, float4 sign_value);

half copysign(half magnitude_value, half sign_value); Added in API level 24

half2 copysign(half2 magnitude_value, half2 sign_value); Added in API level 24

half3 copysign(half3 magnitude_value, half3 sign_value); Added in API level 24

half4 copysign(half4 magnitude_value, half4 sign_value); Added in API level 24

Copies the sign from sign_value to magnitude_value.

The value returned is either magnitude_value or -magnitude_value.

For example, `copysign(4.0f, -2.7f)` returns -4.0f and `copysign(-4.0f, 2.7f)` returns 4.0f.

cos : Cosine

float cos(float v);

float2 cos(float2 v);

float3 cos(float3 v);

float4 cos(float4 v);

half cos(half v); Added in API level 24

half2 cos(half2 v); Added in API level 24

half3 cos(half3 v); Added in API level 24

half4 cos(half4 v); Added in API level 24

Returns the cosine of an angle measured in radians.

See also native_cos().

cosh : Hypebolic cosine

float cosh(float v);

float2 cosh(float2 v);

float3 cosh(float3 v);

float4 cosh(float4 v);

half cosh(half v); Added in API level 24

half2 cosh(half2 v); Added in API level 24

half3 cosh(half3 v); Added in API level 24

half4 cosh(half4 v); Added in API level 24

Returns the hypebolic cosine of v, where v is measured in radians.

See also native_cosh().

cospi : Cosine of a number multiplied by pi

float cospi(float v);

float2 cospi(float2 v);

float3 cospi(float3 v);

float4 cospi(float4 v);

half cospi(half v); Added in API level 24

half2 cospi(half2 v); Added in API level 24

half3 cospi(half3 v); Added in API level 24

half4 cospi(half4 v); Added in API level 24

Returns the cosine of `(v * pi)`, where `(v * pi)` is measured in radians.

To get the cosine of a value measured in degrees, call `cospi(v / 180.f)`.

See also native_cospi().

degrees : Converts radians into degrees

float degrees(float v);

float2 degrees(float2 v);

float3 degrees(float3 v);

float4 degrees(float4 v);

half degrees(half v); Added in API level 24

half2 degrees(half2 v); Added in API level 24

half3 degrees(half3 v); Added in API level 24

half4 degrees(half4 v); Added in API level 24

Converts from radians to degrees.

erf : Mathematical error function

float erf(float v);

float2 erf(float2 v);

float3 erf(float3 v);

float4 erf(float4 v);

half erf(half v); Added in API level 24

half2 erf(half2 v); Added in API level 24

half3 erf(half3 v); Added in API level 24

half4 erf(half4 v); Added in API level 24

Returns the error function.

erfc : Mathematical complementary error function

float erfc(float v);

float2 erfc(float2 v);

float3 erfc(float3 v);

float4 erfc(float4 v);

half erfc(half v); Added in API level 24

half2 erfc(half2 v); Added in API level 24

half3 erfc(half3 v); Added in API level 24

half4 erfc(half4 v); Added in API level 24

Returns the complementary error function.

exp : e raised to a number

float exp(float v);

float2 exp(float2 v);

float3 exp(float3 v);

float4 exp(float4 v);

half exp(half v); Added in API level 24

half2 exp(half2 v); Added in API level 24

half3 exp(half3 v); Added in API level 24

half4 exp(half4 v); Added in API level 24

Returns e raised to v, i.e. e ^ v.

See also native_exp().

exp10 : 10 raised to a number

float exp10(float v);

float2 exp10(float2 v);

float3 exp10(float3 v);

float4 exp10(float4 v);

half exp10(half v); Added in API level 24

half2 exp10(half2 v); Added in API level 24

half3 exp10(half3 v); Added in API level 24

half4 exp10(half4 v); Added in API level 24

Returns 10 raised to v, i.e. 10.f ^ v.

See also native_exp10().

exp2 : 2 raised to a number

float exp2(float v);

float2 exp2(float2 v);

float3 exp2(float3 v);

float4 exp2(float4 v);

half exp2(half v); Added in API level 24

half2 exp2(half2 v); Added in API level 24

half3 exp2(half3 v); Added in API level 24

half4 exp2(half4 v); Added in API level 24

Returns 2 raised to v, i.e. 2.f ^ v.

See also native_exp2().

expm1 : e raised to a number minus one

float expm1(float v);

float2 expm1(float2 v);

float3 expm1(float3 v);

float4 expm1(float4 v);

half expm1(half v); Added in API level 24

half2 expm1(half2 v); Added in API level 24

half3 expm1(half3 v); Added in API level 24

half4 expm1(half4 v); Added in API level 24

Returns e raised to v minus 1, i.e. (e ^ v) - 1.

See also native_expm1().

fabs : Absolute value of a float

float fabs(float v);

float2 fabs(float2 v);

float3 fabs(float3 v);

float4 fabs(float4 v);

half fabs(half v); Added in API level 24

half2 fabs(half2 v); Added in API level 24

half3 fabs(half3 v); Added in API level 24

half4 fabs(half4 v); Added in API level 24

Returns the absolute value of the float v.

For integers, use abs().

fdim : Positive difference between two values

float fdim(float a, float b);

float2 fdim(float2 a, float2 b);

float3 fdim(float3 a, float3 b);

float4 fdim(float4 a, float4 b);

half fdim(half a, half b); Added in API level 24

half2 fdim(half2 a, half2 b); Added in API level 24

half3 fdim(half3 a, half3 b); Added in API level 24

half4 fdim(half4 a, half4 b); Added in API level 24

Returns the positive difference between two values.

If a > b, returns (a - b) otherwise returns 0f.

floor : Smallest integer not greater than a value

float floor(float v);

float2 floor(float2 v);

float3 floor(float3 v);

float4 floor(float4 v);

half floor(half v); Added in API level 24

half2 floor(half2 v); Added in API level 24

half3 floor(half3 v); Added in API level 24

half4 floor(half4 v); Added in API level 24

Returns the smallest integer not greater than a value.

For example, `floor(1.2f)` returns 1.f, and `floor(-1.2f)` returns -2.f.

See also ceil().

fma : Multiply and add

float fma(float multiplicand1, float multiplicand2, float offset);

float2 fma(float2 multiplicand1, float2 multiplicand2, float2 offset);

float3 fma(float3 multiplicand1, float3 multiplicand2, float3 offset);

float4 fma(float4 multiplicand1, float4 multiplicand2, float4 offset);

half fma(half multiplicand1, half multiplicand2, half offset); Added in API level 24

half2 fma(half2 multiplicand1, half2 multiplicand2, half2 offset); Added in API level 24

half3 fma(half3 multiplicand1, half3 multiplicand2, half3 offset); Added in API level 24

half4 fma(half4 multiplicand1, half4 multiplicand2, half4 offset); Added in API level 24

Multiply and add. Returns `(multiplicand1 * multiplicand2) + offset`.

This function is similar to mad(). fma() retains full precision of the multiplied result and rounds only after the addition. mad() rounds after the multiplication and the addition. This extra precision is not guaranteed in rs_fp_relaxed mode.

fmax : Maximum of two floats

float fmax(float a, float b);

float2 fmax(float2 a, float b);

float2 fmax(float2 a, float2 b);

float3 fmax(float3 a, float b);

float3 fmax(float3 a, float3 b);

float4 fmax(float4 a, float b);

float4 fmax(float4 a, float4 b);

half fmax(half a, half b); Added in API level 24

half2 fmax(half2 a, half b); Added in API level 24

half2 fmax(half2 a, half2 b); Added in API level 24

half3 fmax(half3 a, half b); Added in API level 24

half3 fmax(half3 a, half3 b); Added in API level 24

half4 fmax(half4 a, half b); Added in API level 24

half4 fmax(half4 a, half4 b); Added in API level 24

Returns the maximum of a and b, i.e. `(a < b ? b : a)`.

The max() function returns identical results but can be applied to more data types.

fmin : Minimum of two floats

float fmin(float a, float b);

float2 fmin(float2 a, float b);

float2 fmin(float2 a, float2 b);

float3 fmin(float3 a, float b);

float3 fmin(float3 a, float3 b);

float4 fmin(float4 a, float b);

float4 fmin(float4 a, float4 b);

half fmin(half a, half b); Added in API level 24

half2 fmin(half2 a, half b); Added in API level 24

half2 fmin(half2 a, half2 b); Added in API level 24

half3 fmin(half3 a, half b); Added in API level 24

half3 fmin(half3 a, half3 b); Added in API level 24

half4 fmin(half4 a, half b); Added in API level 24

half4 fmin(half4 a, half4 b); Added in API level 24

Returns the minimum of a and b, i.e. `(a > b ? b : a)`.

The min() function returns identical results but can be applied to more data types.

fmod : Modulo

float fmod(float numerator, float denominator);

float2 fmod(float2 numerator, float2 denominator);

float3 fmod(float3 numerator, float3 denominator);

float4 fmod(float4 numerator, float4 denominator);

half fmod(half numerator, half denominator); Added in API level 24

half2 fmod(half2 numerator, half2 denominator); Added in API level 24

half3 fmod(half3 numerator, half3 denominator); Added in API level 24

half4 fmod(half4 numerator, half4 denominator); Added in API level 24

Returns the remainder of (numerator / denominator), where the quotient is rounded towards zero.

The function remainder() is similar but rounds toward the closest interger. For example, `fmod(-3.8f, 2.f)` returns -1.8f (-3.8f - -1.f * 2.f) while `remainder(-3.8f, 2.f)` returns 0.2f (-3.8f - -2.f * 2.f).

fract : Positive fractional part

float fract(float v);

float fract(float v, float* floor);

float2 fract(float2 v);

float2 fract(float2 v, float2* floor);

float3 fract(float3 v);

float3 fract(float3 v, float3* floor);

float4 fract(float4 v);

float4 fract(float4 v, float4* floor);

half fract(half v); Added in API level 24

half fract(half v, half* floor); Added in API level 24

half2 fract(half2 v); Added in API level 24

half2 fract(half2 v, half2* floor); Added in API level 24

half3 fract(half3 v); Added in API level 24

half3 fract(half3 v, half3* floor); Added in API level 24

half4 fract(half4 v); Added in API level 24

half4 fract(half4 v, half4* floor); Added in API level 24

Parameters

v	Input value.
floor	If floor is not null, *floor will be set to the floor of v.

Returns the positive fractional part of v, i.e. `v - floor(v)`.

For example, `fract(1.3f, &val)` returns 0.3f and sets val to 1.f. `fract(-1.3f, &val)` returns 0.7f and sets val to -2.f.

frexp : Binary mantissa and exponent

float frexp(float v, int* exponent);

float2 frexp(float2 v, int2* exponent);

float3 frexp(float3 v, int3* exponent);

float4 frexp(float4 v, int4* exponent);

half frexp(half v, int* exponent); Added in API level 24

half2 frexp(half2 v, int2* exponent); Added in API level 24

half3 frexp(half3 v, int3* exponent); Added in API level 24

half4 frexp(half4 v, int4* exponent); Added in API level 24

Parameters

v	Input value.
exponent	If exponent is not null, *exponent will be set to the exponent of v.

Returns the binary mantissa and exponent of v, i.e. `v == mantissa * 2 ^ exponent`.

The mantissa is always between 0.5 (inclusive) and 1.0 (exclusive).

See ldexp() for the reverse operation. See also logb() and ilogb().

half_recip : Reciprocal computed to 16 bit precision

float half_recip(float v); Added in API level 17

float2 half_recip(float2 v); Added in API level 17

float3 half_recip(float3 v); Added in API level 17

float4 half_recip(float4 v); Added in API level 17

Returns the approximate reciprocal of a value.

The precision is that of a 16 bit floating point value.

See also native_recip().

half_rsqrt : Reciprocal of a square root computed to 16 bit precision

float half_rsqrt(float v); Added in API level 17

float2 half_rsqrt(float2 v); Added in API level 17

float3 half_rsqrt(float3 v); Added in API level 17

float4 half_rsqrt(float4 v); Added in API level 17

Returns the approximate value of `(1.f / sqrt(value))`.

The precision is that of a 16 bit floating point value.

See also rsqrt(), native_rsqrt().

half_sqrt : Square root computed to 16 bit precision

float half_sqrt(float v); Added in API level 17

float2 half_sqrt(float2 v); Added in API level 17

float3 half_sqrt(float3 v); Added in API level 17

float4 half_sqrt(float4 v); Added in API level 17

Returns the approximate square root of a value.

The precision is that of a 16 bit floating point value.

See also sqrt(), native_sqrt().

hypot : Hypotenuse

float hypot(float a, float b);

float2 hypot(float2 a, float2 b);

float3 hypot(float3 a, float3 b);

float4 hypot(float4 a, float4 b);

half hypot(half a, half b); Added in API level 24

half2 hypot(half2 a, half2 b); Added in API level 24

half3 hypot(half3 a, half3 b); Added in API level 24

half4 hypot(half4 a, half4 b); Added in API level 24

Returns the hypotenuse, i.e. `sqrt(a * a + b * b)`.

See also native_hypot().

ilogb : Base two exponent

int ilogb(float v);

int ilogb(half v); Added in API level 24

int2 ilogb(float2 v);

int2 ilogb(half2 v); Added in API level 24

int3 ilogb(float3 v);

int3 ilogb(half3 v); Added in API level 24

int4 ilogb(float4 v);

int4 ilogb(half4 v); Added in API level 24

Returns the base two exponent of a value, where the mantissa is between 1.f (inclusive) and 2.f (exclusive).

For example, `ilogb(8.5f)` returns 3.

Because of the difference in mantissa, this number is one less than is returned by frexp().

logb() is similar but returns a float.

ldexp : Creates a floating point from mantissa and exponent

float ldexp(float mantissa, int exponent);

float2 ldexp(float2 mantissa, int exponent);

float2 ldexp(float2 mantissa, int2 exponent);

float3 ldexp(float3 mantissa, int exponent);

float3 ldexp(float3 mantissa, int3 exponent);

float4 ldexp(float4 mantissa, int exponent);

float4 ldexp(float4 mantissa, int4 exponent);

half ldexp(half mantissa, int exponent); Added in API level 24

half2 ldexp(half2 mantissa, int exponent); Added in API level 24

half2 ldexp(half2 mantissa, int2 exponent); Added in API level 24

half3 ldexp(half3 mantissa, int exponent); Added in API level 24

half3 ldexp(half3 mantissa, int3 exponent); Added in API level 24

half4 ldexp(half4 mantissa, int exponent); Added in API level 24

half4 ldexp(half4 mantissa, int4 exponent); Added in API level 24

Parameters

mantissa Mantissa.

exponent Exponent, a single component or matching vector.

Returns the floating point created from the mantissa and exponent, i.e. (mantissa * 2 ^ exponent).

See frexp() for the reverse operation.

lgamma : Natural logarithm of the gamma function

float lgamma(float v);

float lgamma(float v, int* sign_of_gamma);

float2 lgamma(float2 v);

float2 lgamma(float2 v, int2* sign_of_gamma);

float3 lgamma(float3 v);

float3 lgamma(float3 v, int3* sign_of_gamma);

float4 lgamma(float4 v);

float4 lgamma(float4 v, int4* sign_of_gamma);

half lgamma(half v); Added in API level 24

half lgamma(half v, int* sign_of_gamma); Added in API level 24

half2 lgamma(half2 v); Added in API level 24

half2 lgamma(half2 v, int2* sign_of_gamma); Added in API level 24

half3 lgamma(half3 v); Added in API level 24

half3 lgamma(half3 v, int3* sign_of_gamma); Added in API level 24

half4 lgamma(half4 v); Added in API level 24

half4 lgamma(half4 v, int4* sign_of_gamma); Added in API level 24

Parameters

v

sign_of_gamma If sign_of_gamma is not null, *sign_of_gamma will be set to -1.f if the gamma of v is negative, otherwise to 1.f.

Returns the natural logarithm of the absolute value of the gamma function, i.e. `log(fabs(tgamma(v)))`.

See also tgamma().

log : Natural logarithm

float log(float v);

float2 log(float2 v);

float3 log(float3 v);

float4 log(float4 v);

half log(half v); Added in API level 24

half2 log(half2 v); Added in API level 24

half3 log(half3 v); Added in API level 24

half4 log(half4 v); Added in API level 24

Returns the natural logarithm.

See also native_log().

log10 : Base 10 logarithm

float log10(float v);

float2 log10(float2 v);

float3 log10(float3 v);

float4 log10(float4 v);

half log10(half v); Added in API level 24

half2 log10(half2 v); Added in API level 24

half3 log10(half3 v); Added in API level 24

half4 log10(half4 v); Added in API level 24

Returns the base 10 logarithm.

See also native_log10().

log1p : Natural logarithm of a value plus 1

float log1p(float v);
float2 log1p(float2 v);
float3 log1p(float3 v);
float4 log1p(float4 v);
half log1p(half v); Added in API level 24
half2 log1p(half2 v); Added in API level 24
half3 log1p(half3 v); Added in API level 24
half4 log1p(half4 v); Added in API level 24

Returns the natural logarithm of `(v + 1.f)`.

See also native_log1p().

log2 : Base 2 logarithm

float log2(float v);
float2 log2(float2 v);
float3 log2(float3 v);
float4 log2(float4 v);
half log2(half v); Added in API level 24
half2 log2(half2 v); Added in API level 24
half3 log2(half3 v); Added in API level 24
half4 log2(half4 v); Added in API level 24

Returns the base 2 logarithm.

See also native_log2().

logb : Base two exponent

float logb(float v);
float2 logb(float2 v);
float3 logb(float3 v);
float4 logb(float4 v);
half logb(half v); Added in API level 24
half2 logb(half2 v); Added in API level 24
half3 logb(half3 v); Added in API level 24
half4 logb(half4 v); Added in API level 24

Returns the base two exponent of a value, where the mantissa is between 1.f (inclusive) and 2.f (exclusive).

For example, `logb(8.5f)` returns 3.f.

Because of the difference in mantissa, this number is one less than is returned by frexp().

ilogb() is similar but returns an integer.

mad : Multiply and add

float mad(float multiplicand1, float multiplicand2, float offset);

float2 mad(float2 multiplicand1, float2 multiplicand2, float2 offset);

float3 mad(float3 multiplicand1, float3 multiplicand2, float3 offset);

float4 mad(float4 multiplicand1, float4 multiplicand2, float4 offset);

half mad(half multiplicand1, half multiplicand2, half offset); Added in API level 24

half2 mad(half2 multiplicand1, half2 multiplicand2, half2 offset); Added in API level 24

half3 mad(half3 multiplicand1, half3 multiplicand2, half3 offset); Added in API level 24

half4 mad(half4 multiplicand1, half4 multiplicand2, half4 offset); Added in API level 24

Multiply and add. Returns `(multiplicand1 * multiplicand2) + offset`.

This function is similar to fma(). fma() retains full precision of the multiplied result and rounds only after the addition. mad() rounds after the multiplication and the addition. In rs_fp_relaxed mode, mad() may not do the rounding after multiplicaiton.

max : Maximum

char max(char a, char b);

char2 max(char2 a, char2 b);

char3 max(char3 a, char3 b);

char4 max(char4 a, char4 b);

float max(float a, float b);

float2 max(float2 a, float b);

float2 max(float2 a, float2 b);

float3 max(float3 a, float b);

float3 max(float3 a, float3 b);

float4 max(float4 a, float b);

float4 max(float4 a, float4 b);

half max(half a, half b); Added in API level 24

half2 max(half2 a, half b); Added in API level 24

half2 max(half2 a, half2 b); Added in API level 24

half3 max(half3 a, half b); Added in API level 24

half3 max(half3 a, half3 b); Added in API level 24

half4 max(half4 a, half b); Added in API level 24

half4 max(half4 a, half4 b); Added in API level 24

int max(int a, int b);

int2 max(int2 a, int2 b);

int3 max(int3 a, int3 b);

int4 max(int4 a, int4 b);

long max(long a, long b); Added in API level 21

long2 max(long2 a, long2 b); Added in API level 21

long3 max(long3 a, long3 b); Added in API level 21

long4 max(long4 a, long4 b); Added in API level 21

short max(short a, short b);

short2 max(short2 a, short2 b);

short3 max(short3 a, short3 b);

short4 max(short4 a, short4 b);

uchar max(uchar a, uchar b);

uchar2 max(uchar2 a, uchar2 b);

uchar3 max(uchar3 a, uchar3 b);

uchar4 max(uchar4 a, uchar4 b);

uint max(uint a, uint b);

uint2 max(uint2 a, uint2 b);

uint3 max(uint3 a, uint3 b);

uint4 max(uint4 a, uint4 b);

ulong max(ulong a, ulong b); Added in API level 21

ulong2 max(ulong2 a, ulong2 b); Added in API level 21

ulong3 max(ulong3 a, ulong3 b); Added in API level 21

ulong4 max(ulong4 a, ulong4 b); Added in API level 21

ushort max(ushort a, ushort b);

ushort2 max(ushort2 a, ushort2 b);

ushort3 max(ushort3 a, ushort3 b);

ushort4 max(ushort4 a, ushort4 b);

Returns the maximum value of two arguments.

min : Minimum

char min(char a, char b);

char2 min(char2 a, char2 b);

char3 min(char3 a, char3 b);

char4 min(char4 a, char4 b);

float min(float a, float b);

float2 min(float2 a, float b);

float2 min(float2 a, float2 b);

float3 min(float3 a, float b);

float3 min(float3 a, float3 b);

float4 min(float4 a, float b);

float4 min(float4 a, float4 b);

half min(half a, half b); Added in API level 24

half2 min(half2 a, half b); Added in API level 24

half2 min(half2 a, half2 b); Added in API level 24

half3 min(half3 a, half b); Added in API level 24

half3 min(half3 a, half3 b); Added in API level 24

half4 min(half4 a, half b); Added in API level 24

half4 min(half4 a, half4 b); Added in API level 24

int min(int a, int b);

int2 min(int2 a, int2 b);

int3 min(int3 a, int3 b);

int4 min(int4 a, int4 b);

long min(long a, long b); Added in API level 21

long2 min(long2 a, long2 b); Added in API level 21

long3 min(long3 a, long3 b); Added in API level 21

long4 min(long4 a, long4 b); Added in API level 21

short min(short a, short b);

short2 min(short2 a, short2 b);

short3 min(short3 a, short3 b);

short4 min(short4 a, short4 b);

uchar min(uchar a, uchar b);

uchar2 min(uchar2 a, uchar2 b);

uchar3 min(uchar3 a, uchar3 b);

uchar4 min(uchar4 a, uchar4 b);

uint min(uint a, uint b);

uint2 min(uint2 a, uint2 b);

uint3 min(uint3 a, uint3 b);

uint4 min(uint4 a, uint4 b);

ulong min(ulong a, ulong b); Added in API level 21

ulong2 min(ulong2 a, ulong2 b); Added in API level 21

ulong3 min(ulong3 a, ulong3 b); Added in API level 21

ulong4 min(ulong4 a, ulong4 b); Added in API level 21

ushort min(ushort a, ushort b);

ushort2 min(ushort2 a, ushort2 b);

ushort3 min(ushort3 a, ushort3 b);

ushort4 min(ushort4 a, ushort4 b);

Returns the minimum value of two arguments.

mix : Mixes two values

float mix(float start, float stop, float fraction);

float2 mix(float2 start, float2 stop, float fraction);

float2 mix(float2 start, float2 stop, float2 fraction);

float3 mix(float3 start, float3 stop, float fraction);

float3 mix(float3 start, float3 stop, float3 fraction);

float4 mix(float4 start, float4 stop, float fraction);

float4 mix(float4 start, float4 stop, float4 fraction);

half mix(half start, half stop, half fraction); Added in API level 24

half2 mix(half2 start, half2 stop, half fraction); Added in API level 24

half2 mix(half2 start, half2 stop, half2 fraction); Added in API level 24

half3 mix(half3 start, half3 stop, half fraction); Added in API level 24

half3 mix(half3 start, half3 stop, half3 fraction); Added in API level 24

half4 mix(half4 start, half4 stop, half fraction); Added in API level 24

half4 mix(half4 start, half4 stop, half4 fraction); Added in API level 24

Returns start + ((stop - start) * fraction).

This can be useful for mixing two values. For example, to create a new color that is 40% color1 and 60% color2, use `mix(color1, color2, 0.6f)`.

modf : Integral and fractional components

float modf(float v, float* integral_part);

float2 modf(float2 v, float2* integral_part);

float3 modf(float3 v, float3* integral_part);

float4 modf(float4 v, float4* integral_part);

half modf(half v, half* integral_part); Added in API level 24

half2 modf(half2 v, half2* integral_part); Added in API level 24

half3 modf(half3 v, half3* integral_part); Added in API level 24

half4 modf(half4 v, half4* integral_part); Added in API level 24

Parameters

v	Source value.
integral_part	*integral_part will be set to the integral portion of the number.

Returns

Floating point portion of the value.

Returns the integral and fractional components of a number.

Both components will have the same sign as x. For example, for an input of -3.72f, *integral_part will be set to -3.f and .72f will be returned.

nan : Not a Number

float nan(uint v);

Parameters

v	Not used.

Returns a NaN value (Not a Number).

nan_half : Not a Number

half nan_half(); Added in API level 24

Returns a half-precision floating point NaN value (Not a Number).

native_acos : Approximate inverse cosine

float native_acos(float v);	Added in API level 21
float2 native_acos(float2 v);	Added in API level 21
float3 native_acos(float3 v);	Added in API level 21
float4 native_acos(float4 v);	Added in API level 21
half native_acos(half v);	Added in API level 24
half2 native_acos(half2 v);	Added in API level 24
half3 native_acos(half3 v);	Added in API level 24
half4 native_acos(half4 v);	Added in API level 24

Returns the approximate inverse cosine, in radians.

This function yields undefined results from input values less than -1 or greater than 1.

See also acos().

native_acosh : Approximate inverse hyperbolic cosine

float native_acosh(float v);	Added in API level 21
float2 native_acosh(float2 v);	Added in API level 21
float3 native_acosh(float3 v);	Added in API level 21
float4 native_acosh(float4 v);	Added in API level 21
half native_acosh(half v);	Added in API level 24
half2 native_acosh(half2 v);	Added in API level 24
half3 native_acosh(half3 v);	Added in API level 24
half4 native_acosh(half4 v);	Added in API level 24

Returns the approximate inverse hyperbolic cosine, in radians.

See also acosh().

native_acospi : Approximate inverse cosine divided by pi

float native_acospi(float v);	Added in API level 21
float2 native_acospi(float2 v);	Added in API level 21
float3 native_acospi(float3 v);	Added in API level 21
float4 native_acospi(float4 v);	Added in API level 21
half native_acospi(half v);	Added in API level 24
half2 native_acospi(half2 v);	Added in API level 24
half3 native_acospi(half3 v);	Added in API level 24
half4 native_acospi(half4 v);	Added in API level 24

Returns the approximate inverse cosine in radians, divided by pi.

To get an inverse cosine measured in degrees, use `acospi(a) * 180.f`.

This function yields undefined results from input values less than -1 or greater than 1.

See also acospi().

native_asin : Approximate inverse sine

float native_asin(float v);	Added in API level 21
float2 native_asin(float2 v);	Added in API level 21
float3 native_asin(float3 v);	Added in API level 21
float4 native_asin(float4 v);	Added in API level 21
half native_asin(half v);	Added in API level 24
half2 native_asin(half2 v);	Added in API level 24
half3 native_asin(half3 v);	Added in API level 24
half4 native_asin(half4 v);	Added in API level 24

Returns the approximate inverse sine, in radians.

This function yields undefined results from input values less than -1 or greater than 1.

See also asin().

native_asinh : Approximate inverse hyperbolic sine

float native_asinh(float v);	Added in API level 21
float2 native_asinh(float2 v);	Added in API level 21
float3 native_asinh(float3 v);	Added in API level 21
float4 native_asinh(float4 v);	Added in API level 21
half native_asinh(half v);	Added in API level 24
half2 native_asinh(half2 v);	Added in API level 24
half3 native_asinh(half3 v);	Added in API level 24
half4 native_asinh(half4 v);	Added in API level 24

Returns the approximate inverse hyperbolic sine, in radians.

See also asinh().

native_asinpi : Approximate inverse sine divided by pi

float native_asinpi(float v);	Added in API level 21
float2 native_asinpi(float2 v);	Added in API level 21

float3 native_asinpi(float3 v);	Added in API level 21
float4 native_asinpi(float4 v);	Added in API level 21
half native_asinpi(half v);	Added in API level 24
half2 native_asinpi(half2 v);	Added in API level 24
half3 native_asinpi(half3 v);	Added in API level 24
half4 native_asinpi(half4 v);	Added in API level 24

Returns the approximate inverse sine in radians, divided by pi.

To get an inverse sine measured in degrees, use `asinpi(a) * 180.f`.

This function yields undefined results from input values less than -1 or greater than 1.

See also asinpi().

native_atan : Approximate inverse tangent

float native_atan(float v);	Added in API level 21
float2 native_atan(float2 v);	Added in API level 21
float3 native_atan(float3 v);	Added in API level 21
float4 native_atan(float4 v);	Added in API level 21
half native_atan(half v);	Added in API level 24
half2 native_atan(half2 v);	Added in API level 24
half3 native_atan(half3 v);	Added in API level 24
half4 native_atan(half4 v);	Added in API level 24

Returns the approximate inverse tangent, in radians.

See also atan().

native_atan2 : Approximate inverse tangent of a ratio

float native_atan2(float numerator, float denominator);	Added in API level 21
float2 native_atan2(float2 numerator, float2 denominator);	Added in API level 21
float3 native_atan2(float3 numerator, float3 denominator);	Added in API level 21
float4 native_atan2(float4 numerator, float4 denominator);	Added in API level 21
half native_atan2(half numerator, half denominator);	Added in API level 24
half2 native_atan2(half2 numerator, half2 denominator);	Added in API level 24
half3 native_atan2(half3 numerator, half3 denominator);	Added in API level 24
half4 native_atan2(half4 numerator, half4 denominator);	Added in API level 24

Parameters

numerator	Numerator.
denominator	Denominator. Can be 0.

Returns the approximate inverse tangent of `(numerator / denominator)`, in radians.

See also atan2().

native_atan2pi : Approximate inverse tangent of a ratio, divided by pi

float native_atan2pi(float numerator, float denominator);	Added in API level 21
float2 native_atan2pi(float2 numerator, float2 denominator);	Added in API level 21
float3 native_atan2pi(float3 numerator, float3 denominator);	Added in API level 21
float4 native_atan2pi(float4 numerator, float4 denominator);	Added in API level 21
half native_atan2pi(half numerator, half denominator);	Added in API level 24

half2 native_atan2pi(half2 numerator, half2 denominator); Added in API level 24

half3 native_atan2pi(half3 numerator, half3 denominator); Added in API level 24

half4 native_atan2pi(half4 numerator, half4 denominator); Added in API level 24

Parameters

numerator Numerator.

denominator Denominator. Can be 0.

Returns the approximate inverse tangent of `(numerator / denominator)`, in radians, divided by pi.

To get an inverse tangent measured in degrees, use `atan2pi(n, d) * 180.f`.

See also atan2pi().

native_atanh : Approximate inverse hyperbolic tangent

float native_atanh(float v); Added in API level 21

float2 native_atanh(float2 v); Added in API level 21

float3 native_atanh(float3 v); Added in API level 21

float4 native_atanh(float4 v); Added in API level 21

half native_atanh(half v); Added in API level 24

half2 native_atanh(half2 v); Added in API level 24

half3 native_atanh(half3 v); Added in API level 24

half4 native_atanh(half4 v); Added in API level 24

Returns the approximate inverse hyperbolic tangent, in radians.

See also atanh().

native_atanpi : Approximate inverse tangent divided by pi

float native_atanpi(float v); Added in API level 21

float2 native_atanpi(float2 v); Added in API level 21

float3 native_atanpi(float3 v); Added in API level 21

float4 native_atanpi(float4 v); Added in API level 21

half native_atanpi(half v); Added in API level 24

half2 native_atanpi(half2 v); Added in API level 24

half3 native_atanpi(half3 v); Added in API level 24

half4 native_atanpi(half4 v); Added in API level 24

Returns the approximate inverse tangent in radians, divided by pi.

To get an inverse tangent measured in degrees, use `atanpi(a) * 180.f`.

See also atanpi().

native_cbrt : Approximate cube root

float native_cbrt(float v); Added in API level 21

float2 native_cbrt(float2 v); Added in API level 21

float3 native_cbrt(float3 v); Added in API level 21

float4 native_cbrt(float4 v); Added in API level 21

half native_cbrt(half v); Added in API level 24

half2 native_cbrt(half2 v); Added in API level 24

half3 native_cbrt(half3 v); Added in API level 24

half4 native_cbrt(half4 v); Added in API level 24

Returns the approximate cubic root.

See also cbrt().

native_cos : Approximate cosine

float native_cos(float v);	Added in API level 21
float2 native_cos(float2 v);	Added in API level 21
float3 native_cos(float3 v);	Added in API level 21
float4 native_cos(float4 v);	Added in API level 21
half native_cos(half v);	Added in API level 24
half2 native_cos(half2 v);	Added in API level 24
half3 native_cos(half3 v);	Added in API level 24
half4 native_cos(half4 v);	Added in API level 24

Returns the approximate cosine of an angle measured in radians.

See also cos().

native_cosh : Approximate hypebolic cosine

float native_cosh(float v);	Added in API level 21
float2 native_cosh(float2 v);	Added in API level 21
float3 native_cosh(float3 v);	Added in API level 21
float4 native_cosh(float4 v);	Added in API level 21
half native_cosh(half v);	Added in API level 24
half2 native_cosh(half2 v);	Added in API level 24
half3 native_cosh(half3 v);	Added in API level 24
half4 native_cosh(half4 v);	Added in API level 24

Returns the approximate hypebolic cosine.

See also cosh().

native_cospi : Approximate cosine of a number multiplied by pi

float native_cospi(float v);	Added in API level 21
float2 native_cospi(float2 v);	Added in API level 21
float3 native_cospi(float3 v);	Added in API level 21
float4 native_cospi(float4 v);	Added in API level 21
half native_cospi(half v);	Added in API level 24
half2 native_cospi(half2 v);	Added in API level 24
half3 native_cospi(half3 v);	Added in API level 24
half4 native_cospi(half4 v);	Added in API level 24

Returns the approximate cosine of (v * pi), where (v * pi) is measured in radians.

To get the cosine of a value measured in degrees, call `cospi(v / 180.f)`.

See also cospi().

native_divide : Approximate division

float native_divide(float left_vector, float right_vector);	Added in API level 21
float2 native_divide(float2 left_vector, float2 right_vector);	Added in API level 21

float3 native_divide(float3 left_vector, float3 right_vector);	Added in API level 21
float4 native_divide(float4 left_vector, float4 right_vector);	Added in API level 21
half native_divide(half left_vector, half right_vector);	Added in API level 24
half2 native_divide(half2 left_vector, half2 right_vector);	Added in API level 24
half3 native_divide(half3 left_vector, half3 right_vector);	Added in API level 24
half4 native_divide(half4 left_vector, half4 right_vector);	Added in API level 24

Computes the approximate division of two values.

native_exp : Approximate e raised to a number

float native_exp(float v);	Added in API level 18
float2 native_exp(float2 v);	Added in API level 18
float3 native_exp(float3 v);	Added in API level 18
float4 native_exp(float4 v);	Added in API level 18
half native_exp(half v);	Added in API level 24
half2 native_exp(half2 v);	Added in API level 24
half3 native_exp(half3 v);	Added in API level 24
half4 native_exp(half4 v);	Added in API level 24

Fast approximate exp.

It is valid for inputs from -86.f to 86.f. The precision is no worse than what would be expected from using 16 bit floating point values.

See also exp().

native_exp10 : Approximate 10 raised to a number

float native_exp10(float v);	Added in API level 18
float2 native_exp10(float2 v);	Added in API level 18
float3 native_exp10(float3 v);	Added in API level 18
float4 native_exp10(float4 v);	Added in API level 18
half native_exp10(half v);	Added in API level 24
half2 native_exp10(half2 v);	Added in API level 24
half3 native_exp10(half3 v);	Added in API level 24
half4 native_exp10(half4 v);	Added in API level 24

Fast approximate exp10.

It is valid for inputs from -37.f to 37.f. The precision is no worse than what would be expected from using 16 bit floating point values.

See also exp10().

native_exp2 : Approximate 2 raised to a number

float native_exp2(float v);	Added in API level 18
float2 native_exp2(float2 v);	Added in API level 18
float3 native_exp2(float3 v);	Added in API level 18
float4 native_exp2(float4 v);	Added in API level 18
half native_exp2(half v);	Added in API level 24
half2 native_exp2(half2 v);	Added in API level 24
half3 native_exp2(half3 v);	Added in API level 24
half4 native_exp2(half4 v);	Added in API level 24

Fast approximate exp2.

It is valid for inputs from -125.f to 125.f. The precision is no worse than what would be expected from using 16 bit floating point values.

See also exp2().

native_expm1 : Approximate e raised to a number minus one

float native_expm1(float v);	Added in API level 21
float2 native_expm1(float2 v);	Added in API level 21
float3 native_expm1(float3 v);	Added in API level 21
float4 native_expm1(float4 v);	Added in API level 21
half native_expm1(half v);	Added in API level 24
half2 native_expm1(half2 v);	Added in API level 24
half3 native_expm1(half3 v);	Added in API level 24
half4 native_expm1(half4 v);	Added in API level 24

Returns the approximate (e ^ v) - 1.

See also expm1().

native_hypot : Approximate hypotenuse

float native_hypot(float a, float b);	Added in API level 21
float2 native_hypot(float2 a, float2 b);	Added in API level 21
float3 native_hypot(float3 a, float3 b);	Added in API level 21
float4 native_hypot(float4 a, float4 b);	Added in API level 21
half native_hypot(half a, half b);	Added in API level 24
half2 native_hypot(half2 a, half2 b);	Added in API level 24
half3 native_hypot(half3 a, half3 b);	Added in API level 24
half4 native_hypot(half4 a, half4 b);	Added in API level 24

Returns the approximate native_sqrt(a * a + b * b)

See also hypot().

native_log : Approximate natural logarithm

float native_log(float v);	Added in API level 18
float2 native_log(float2 v);	Added in API level 18
float3 native_log(float3 v);	Added in API level 18
float4 native_log(float4 v);	Added in API level 18
half native_log(half v);	Added in API level 24
half2 native_log(half2 v);	Added in API level 24
half3 native_log(half3 v);	Added in API level 24
half4 native_log(half4 v);	Added in API level 24

Fast approximate log.

It is not accurate for values very close to zero.

See also log().

native_log10 : Approximate base 10 logarithm

float native_log10(float v);	Added in API level 18
float2 native_log10(float2 v);	Added in API level 18

float3 native_log10(float3 v); Added in API level 18

float4 native_log10(float4 v); Added in API level 18

half native_log10(half v); Added in API level 24

half2 native_log10(half2 v); Added in API level 24

half3 native_log10(half3 v); Added in API level 24

half4 native_log10(half4 v); Added in API level 24

Fast approximate log10.

It is not accurate for values very close to zero.

See also log10().

native_log1p : Approximate natural logarithm of a value plus 1

float native_log1p(float v); Added in API level 21

float2 native_log1p(float2 v); Added in API level 21

float3 native_log1p(float3 v); Added in API level 21

float4 native_log1p(float4 v); Added in API level 21

half native_log1p(half v); Added in API level 24

half2 native_log1p(half2 v); Added in API level 24

half3 native_log1p(half3 v); Added in API level 24

half4 native_log1p(half4 v); Added in API level 24

Returns the approximate natural logarithm of (v + 1.0f)

See also log1p().

native_log2 : Approximate base 2 logarithm

float native_log2(float v); Added in API level 18

float2 native_log2(float2 v); Added in API level 18

float3 native_log2(float3 v); Added in API level 18

float4 native_log2(float4 v); Added in API level 18

half native_log2(half v); Added in API level 24

half2 native_log2(half2 v); Added in API level 24

half3 native_log2(half3 v); Added in API level 24

half4 native_log2(half4 v); Added in API level 24

Fast approximate log2.

It is not accurate for values very close to zero.

See also log2().

native_powr : Approximate positive base raised to an exponent

float native_powr(float base, float exponent); Added in API level 18

float2 native_powr(float2 base, float2 exponent); Added in API level 18

float3 native_powr(float3 base, float3 exponent); Added in API level 18

float4 native_powr(float4 base, float4 exponent); Added in API level 18

half native_powr(half base, half exponent); Added in API level 24

half2 native_powr(half2 base, half2 exponent); Added in API level 24

half3 native_powr(half3 base, half3 exponent); Added in API level 24

half4 native_powr(half4 base, half4 exponent); Added in API level 24

Parameters

base Must be between 0.f and 256.f. The function is not accurate for values very close to zero.

exponent Must be between -15.f and 15.f.

Fast approximate (base ^ exponent).

See also powr().

native_recip : Approximate reciprocal

float native_recip(float v); Added in API level 21

float2 native_recip(float2 v); Added in API level 21

float3 native_recip(float3 v); Added in API level 21

float4 native_recip(float4 v); Added in API level 21

half native_recip(half v); Added in API level 24

half2 native_recip(half2 v); Added in API level 24

half3 native_recip(half3 v); Added in API level 24

half4 native_recip(half4 v); Added in API level 24

Returns the approximate approximate reciprocal of a value.

See also half_recip().

native_rootn : Approximate nth root

float native_rootn(float v, int n); Added in API level 21

float2 native_rootn(float2 v, int2 n); Added in API level 21

float3 native_rootn(float3 v, int3 n); Added in API level 21

float4 native_rootn(float4 v, int4 n); Added in API level 21

half native_rootn(half v, int n); Added in API level 24

half2 native_rootn(half2 v, int2 n); Added in API level 24

half3 native_rootn(half3 v, int3 n); Added in API level 24

half4 native_rootn(half4 v, int4 n); Added in API level 24

Compute the approximate Nth root of a value.

See also rootn().

native_rsqrt : Approximate reciprocal of a square root

float native_rsqrt(float v); Added in API level 21

float2 native_rsqrt(float2 v); Added in API level 21

float3 native_rsqrt(float3 v); Added in API level 21

float4 native_rsqrt(float4 v); Added in API level 21

half native_rsqrt(half v); Added in API level 24

half2 native_rsqrt(half2 v); Added in API level 24

half3 native_rsqrt(half3 v); Added in API level 24

half4 native_rsqrt(half4 v); Added in API level 24

Returns approximate (1 / sqrt(v)).

See also rsqrt(), half_rsqrt().

native_sin : Approximate sine

float native_sin(float v); Added in API level 21

float2 native_sin(float2 v); Added in API level 21
float3 native_sin(float3 v); Added in API level 21
float4 native_sin(float4 v); Added in API level 21
half native_sin(half v); Added in API level 24
half2 native_sin(half2 v); Added in API level 24
half3 native_sin(half3 v); Added in API level 24
half4 native_sin(half4 v); Added in API level 24

Returns the approximate sine of an angle measured in radians.

See also sin().

native_sincos : Approximate sine and cosine

float native_sincos(float v, float* cos); Added in API level 21
float2 native_sincos(float2 v, float2* cos); Added in API level 21
float3 native_sincos(float3 v, float3* cos); Added in API level 21
float4 native_sincos(float4 v, float4* cos); Added in API level 21
half native_sincos(half v, half* cos); Added in API level 24
half2 native_sincos(half2 v, half2* cos); Added in API level 24
half3 native_sincos(half3 v, half3* cos); Added in API level 24
half4 native_sincos(half4 v, half4* cos); Added in API level 24

Parameters

v Incoming value in radians.

cos *cos will be set to the cosine value.

Returns

Sine.

Returns the approximate sine and cosine of a value.

See also sincos().

native_sinh : Approximate hyperbolic sine

float native_sinh(float v); Added in API level 21
float2 native_sinh(float2 v); Added in API level 21
float3 native_sinh(float3 v); Added in API level 21
float4 native_sinh(float4 v); Added in API level 21
half native_sinh(half v); Added in API level 24
half2 native_sinh(half2 v); Added in API level 24
half3 native_sinh(half3 v); Added in API level 24
half4 native_sinh(half4 v); Added in API level 24

Returns the approximate hyperbolic sine of a value specified in radians.

See also sinh().

native_sinpi : Approximate sine of a number multiplied by pi

float native_sinpi(float v); Added in API level 21
float2 native_sinpi(float2 v); Added in API level 21
float3 native_sinpi(float3 v); Added in API level 21

float4 native_sinpi(float4 v);	Added in API level 21
half native_sinpi(half v);	Added in API level 24
half2 native_sinpi(half2 v);	Added in API level 24
half3 native_sinpi(half3 v);	Added in API level 24
half4 native_sinpi(half4 v);	Added in API level 24

Returns the approximate sine of (v * pi), where (v * pi) is measured in radians.

To get the sine of a value measured in degrees, call `sinpi(v / 180.f)`.

See also sinpi().

native_sqrt : Approximate square root

float native_sqrt(float v);	Added in API level 21
float2 native_sqrt(float2 v);	Added in API level 21
float3 native_sqrt(float3 v);	Added in API level 21
float4 native_sqrt(float4 v);	Added in API level 21
half native_sqrt(half v);	Added in API level 24
half2 native_sqrt(half2 v);	Added in API level 24
half3 native_sqrt(half3 v);	Added in API level 24
half4 native_sqrt(half4 v);	Added in API level 24

Returns the approximate sqrt(v).

See also sqrt(), half_sqrt().

native_tan : Approximate tangent

float native_tan(float v);	Added in API level 21
float2 native_tan(float2 v);	Added in API level 21
float3 native_tan(float3 v);	Added in API level 21
float4 native_tan(float4 v);	Added in API level 21
half native_tan(half v);	Added in API level 24
half2 native_tan(half2 v);	Added in API level 24
half3 native_tan(half3 v);	Added in API level 24
half4 native_tan(half4 v);	Added in API level 24

Returns the approximate tangent of an angle measured in radians.

native_tanh : Approximate hyperbolic tangent

float native_tanh(float v);	Added in API level 21
float2 native_tanh(float2 v);	Added in API level 21
float3 native_tanh(float3 v);	Added in API level 21
float4 native_tanh(float4 v);	Added in API level 21
half native_tanh(half v);	Added in API level 24
half2 native_tanh(half2 v);	Added in API level 24
half3 native_tanh(half3 v);	Added in API level 24
half4 native_tanh(half4 v);	Added in API level 24

Returns the approximate hyperbolic tangent of a value.

See also tanh().

native_tanpi : Approximate tangent of a number multiplied by pi

float native_tanpi(float v); Added in API level 21

float2 native_tanpi(float2 v); Added in API level 21

float3 native_tanpi(float3 v); Added in API level 21

float4 native_tanpi(float4 v); Added in API level 21

half native_tanpi(half v); Added in API level 24

half2 native_tanpi(half2 v); Added in API level 24

half3 native_tanpi(half3 v); Added in API level 24

half4 native_tanpi(half4 v); Added in API level 24

Returns the approximate tangent of (v * pi), where (v * pi) is measured in radians.

To get the tangent of a value measured in degrees, call `tanpi(v / 180.f)`.

See also tanpi().

nextafter : Next floating point number

float nextafter(float v, float target);

float2 nextafter(float2 v, float2 target);

float3 nextafter(float3 v, float3 target);

float4 nextafter(float4 v, float4 target);

half nextafter(half v, half target); Added in API level 24

half2 nextafter(half2 v, half2 target); Added in API level 24

half3 nextafter(half3 v, half3 target); Added in API level 24

half4 nextafter(half4 v, half4 target); Added in API level 24

Returns the next representable floating point number from v towards target.

In rs_fp_relaxed mode, a denormalized input value may not yield the next denormalized value, as support of denormalized values is optional in relaxed mode.

pow : Base raised to an exponent

float pow(float base, float exponent);

float2 pow(float2 base, float2 exponent);

float3 pow(float3 base, float3 exponent);

float4 pow(float4 base, float4 exponent);

half pow(half base, half exponent); Added in API level 24

half2 pow(half2 base, half2 exponent); Added in API level 24

half3 pow(half3 base, half3 exponent); Added in API level 24

half4 pow(half4 base, half4 exponent); Added in API level 24

Returns base raised to the power exponent, i.e. base ^ exponent.

pown() and powr() are similar. pown() takes an integer exponent. powr() assumes the base to be non-negative.

pown : Base raised to an integer exponent

float pown(float base, int exponent);

float2 pown(float2 base, int2 exponent);

float3 pown(float3 base, int3 exponent);

float4 pown(float4 base, int4 exponent);

half pown(half base, int exponent); Added in API level 24
half2 pown(half2 base, int2 exponent); Added in API level 24

half3 pown(half3 base, int3 exponent); Added in API level 24

half4 pown(half4 base, int4 exponent); Added in API level 24

Returns base raised to the power exponent, i.e. base ^ exponent.

pow() and powr() are similar. The both take a float exponent. powr() also assumes the base to be non-negative.

powr : Positive base raised to an exponent

float powr(float base, float exponent);

float2 powr(float2 base, float2 exponent);

float3 powr(float3 base, float3 exponent);

float4 powr(float4 base, float4 exponent);

half powr(half base, half exponent); Added in API level 24

half2 powr(half2 base, half2 exponent); Added in API level 24

half3 powr(half3 base, half3 exponent); Added in API level 24

half4 powr(half4 base, half4 exponent); Added in API level 24

Returns base raised to the power exponent, i.e. base ^ exponent. base must be >= 0.

pow() and pown() are similar. They both make no assumptions about the base. pow() takes a float exponent while pown() take an integer.

See also native_powr().

radians : Converts degrees into radians

float radians(float v);

float2 radians(float2 v);

float3 radians(float3 v);

float4 radians(float4 v);

half radians(half v); Added in API level 24

half2 radians(half2 v); Added in API level 24

half3 radians(half3 v); Added in API level 24

half4 radians(half4 v); Added in API level 24

Converts from degrees to radians.

remainder : Remainder of a division

float remainder(float numerator, float denominator);

float2 remainder(float2 numerator, float2 denominator);

float3 remainder(float3 numerator, float3 denominator);

float4 remainder(float4 numerator, float4 denominator);

half remainder(half numerator, half denominator); Added in API level 24

half2 remainder(half2 numerator, half2 denominator); Added in API level 24

half3 remainder(half3 numerator, half3 denominator); Added in API level 24

half4 remainder(half4 numerator, half4 denominator); Added in API level 24

Returns the remainder of (numerator / denominator), where the quotient is rounded towards the nearest integer.

The function fmod() is similar but rounds toward the closest interger. For example, `fmod(-3.8f, 2.f)` returns -1.8f (-3.8f - -1.f * 2.f) while `remainder(-3.8f, 2.f)` returns 0.2f (-3.8f - -2.f * 2.f).

remquo : Remainder and quotient of a division

float remquo(float numerator, float denominator, int* quotient);

float2 remquo(float2 numerator, float2 denominator, int2* quotient);

float3 remquo(float3 numerator, float3 denominator, int3* quotient);

float4 remquo(float4 numerator, float4 denominator, int4* quotient);

half remquo(half numerator, half denominator, int* quotient); Added in API level 24

half2 remquo(half2 numerator, half2 denominator, int2* quotient); Added in API level 24

half3 remquo(half3 numerator, half3 denominator, int3* quotient); Added in API level 24

half4 remquo(half4 numerator, half4 denominator, int4* quotient); Added in API level 24

Parameters

numerator	Numerator.
denominator	Denominator.
quotient	*quotient will be set to the integer quotient.

Returns

Remainder, precise only for the low three bits.

Returns the quotient and the remainder of (numerator / denominator).

Only the sign and lowest three bits of the quotient are guaranteed to be accurate.

This function is useful for implementing periodic functions. The low three bits of the quotient gives the quadrant and the remainder the distance within the quadrant. For example, an implementation of sin(x) could call `remquo(x, PI / 2.f, &quadrant)` to reduce very large value of x to something within a limited range.

Example: `remquo(-23.5f, 8.f, &quot)` sets the lowest three bits of quot to 3 and the sign negative. It returns 0.5f.

rint : Round to even

float rint(float v);

float2 rint(float2 v);

float3 rint(float3 v);

float4 rint(float4 v);

half rint(half v); Added in API level 24

half2 rint(half2 v); Added in API level 24

half3 rint(half3 v); Added in API level 24

half4 rint(half4 v); Added in API level 24

Rounds to the nearest integral value.

rint() rounds half values to even. For example, `rint(0.5f)` returns 0.f and `rint(1.5f)` returns 2.f. Similarly, `rint(-0.5f)` returns -0.f and `rint(-1.5f)` returns -2.f.

round() is similar but rounds away from zero. trunc() truncates the decimal fraction.

rootn : Nth root

float rootn(float v, int n);

float2 rootn(float2 v, int2 n);

float3 rootn(float3 v, int3 n);

float4 rootn(float4 v, int4 n);

half rootn(half v, int n); Added in API level 24

half2 rootn(half2 v, int2 n); Added in API level 24

half3 rootn(half3 v, int3 n); Added in API level 24

half4 rootn(half4 v, int4 n); Added in API level 24

Compute the Nth root of a value.

See also native_rootn().

round : Round away from zero

float round(float v);

float2 round(float2 v);

float3 round(float3 v);

float4 round(float4 v);

half round(half v); Added in API level 24

half2 round(half2 v); Added in API level 24

half3 round(half3 v); Added in API level 24

half4 round(half4 v); Added in API level 24

Round to the nearest integral value.

round() rounds half values away from zero. For example, `round(0.5f)` returns 1.f and `round(1.5f)` returns 2.f. Similarly, `round(-0.5f)` returns -1.f and `round(-1.5f)` returns -2.f.

rint() is similar but rounds half values toward even. trunc() truncates the decimal fraction.

rsClamp : Restrain a value to a range

char rsClamp(char amount, char low, char high);

int rsClamp(int amount, int low, int high);

short rsClamp(short amount, short low, short high);

uchar rsClamp(uchar amount, uchar low, uchar high);

uint rsClamp(uint amount, uint low, uint high);

ushort rsClamp(ushort amount, ushort low, ushort high);

Parameters

amount	Value to clamp.
low	Lower bound.
high	Upper bound.

Deprecated. Use clamp() instead.

Clamp a value between low and high.

rsFrac : Returns the fractional part of a float

float rsFrac(float v);

Deprecated. Use fract() instead.

Returns the fractional part of a float

rsRand : Pseudo-random number

float rsRand(float max_value);

float rsRand(float min_value, float max_value);

int rsRand(int max_value);

int rsRand(int min_value, int max_value);

Return a random value between 0 (or min_value) and max_malue.

rsqrt : Reciprocal of a square root

float rsqrt(float v);

float2 rsqrt(float2 v);

float3 rsqrt(float3 v);

float4 rsqrt(float4 v);

half rsqrt(half v); Added in API level 24

half2 rsqrt(half2 v); Added in API level 24

half3 rsqrt(half3 v); Added in API level 24

half4 rsqrt(half4 v); Added in API level 24

Returns (1 / sqrt(v)).

See also half_rsqrt(), native_rsqrt().

sign : Sign of a value

float sign(float v);

float2 sign(float2 v);

float3 sign(float3 v);

float4 sign(float4 v);

half sign(half v); Added in API level 24

half2 sign(half2 v); Added in API level 24

half3 sign(half3 v); Added in API level 24

half4 sign(half4 v); Added in API level 24

Returns the sign of a value.

if (v < 0) return -1.f; else if (v > 0) return 1.f; else return 0.f;

sin : Sine

float sin(float v);

float2 sin(float2 v);

float3 sin(float3 v);

float4 sin(float4 v);

half sin(half v); Added in API level 24

half2 sin(half2 v); Added in API level 24

half3 sin(half3 v); Added in API level 24

half4 sin(half4 v); Added in API level 24

Returns the sine of an angle measured in radians.

See also native_sin().

sincos : Sine and cosine

float sincos(float v, float* cos);

float2 sincos(float2 v, float2* cos);

float3 sincos(float3 v, float3* cos);

float4 sincos(float4 v, float4* cos);

half sincos(half v, half* cos); Added in API level 24

half2 sincos(half2 v, half2* cos); Added in API level 24

half3 sincos(half3 v, half3* cos); Added in API level 24

half4 sincos(half4 v, half4* cos); Added in API level 24

Parameters

v Incoming value in radians.

cos *cos will be set to the cosine value.

Returns

Sine of v.

Returns the sine and cosine of a value.

See also native_sincos().

sinh : Hyperbolic sine

float sinh(float v);

float2 sinh(float2 v);

float3 sinh(float3 v);

float4 sinh(float4 v);

half sinh(half v); Added in API level 24

half2 sinh(half2 v); Added in API level 24

half3 sinh(half3 v); Added in API level 24

half4 sinh(half4 v); Added in API level 24

Returns the hyperbolic sine of v, where v is measured in radians.

See also native_sinh().

sinpi : Sine of a number multiplied by pi

float sinpi(float v);

float2 sinpi(float2 v);

float3 sinpi(float3 v);

float4 sinpi(float4 v);

half sinpi(half v); Added in API level 24

half2 sinpi(half2 v); Added in API level 24

half3 sinpi(half3 v); Added in API level 24

half4 sinpi(half4 v); Added in API level 24

Returns the sine of (v * pi), where (v * pi) is measured in radians.

To get the sine of a value measured in degrees, call `sinpi(v / 180.f)`.

See also native_sinpi().

sqrt : Square root

float sqrt(float v);

float2 sqrt(float2 v);

float3 sqrt(float3 v);

float4 sqrt(float4 v);

half sqrt(half v); Added in API level 24

half2 sqrt(half2 v); Added in API level 24

half3 sqrt(half3 v); Added in API level 24

half4 sqrt(half4 v); Added in API level 24

Returns the square root of a value.

See also half_sqrt(), native_sqrt().

step : 0 if less than a value, 0 otherwise

float step(float edge, float v);
float2 step(float edge, float2 v); Added in API level 21
float2 step(float2 edge, float v);
float2 step(float2 edge, float2 v);
float3 step(float edge, float3 v); Added in API level 21
float3 step(float3 edge, float v);
float3 step(float3 edge, float3 v);
float4 step(float edge, float4 v); Added in API level 21
float4 step(float4 edge, float v);
float4 step(float4 edge, float4 v);
half step(half edge, half v); Added in API level 24
half2 step(half edge, half2 v); Added in API level 24
half2 step(half2 edge, half v); Added in API level 24
half2 step(half2 edge, half2 v); Added in API level 24
half3 step(half edge, half3 v); Added in API level 24
half3 step(half3 edge, half v); Added in API level 24
half3 step(half3 edge, half3 v); Added in API level 24
half4 step(half edge, half4 v); Added in API level 24
half4 step(half4 edge, half v); Added in API level 24
half4 step(half4 edge, half4 v); Added in API level 24

Returns 0.f if v < edge, 1.f otherwise.

This can be useful to create conditional computations without using loops and branching instructions. For example, instead of computing `(a[i] < b[i]) ? 0.f : atan2(a[i], b[i])` for the corresponding elements of a vector, you could instead use `step(a, b) * atan2(a, b)`.

tan : Tangent

float tan(float v);
float2 tan(float2 v);
float3 tan(float3 v);
float4 tan(float4 v);
half tan(half v); Added in API level 24
half2 tan(half2 v); Added in API level 24
half3 tan(half3 v); Added in API level 24
half4 tan(half4 v); Added in API level 24

Returns the tangent of an angle measured in radians.

See also native_tan().

tanh : Hyperbolic tangent

float tanh(float v);
float2 tanh(float2 v);
float3 tanh(float3 v);

float4 tanh(float4 v);

half tanh(half v); Added in API level 24

half2 tanh(half2 v); Added in API level 24

half3 tanh(half3 v); Added in API level 24

half4 tanh(half4 v); Added in API level 24

Returns the hyperbolic tangent of a value.

See also native_tanh().

tanpi : Tangent of a number multiplied by pi

float tanpi(float v);

float2 tanpi(float2 v);

float3 tanpi(float3 v);

float4 tanpi(float4 v);

half tanpi(half v); Added in API level 24

half2 tanpi(half2 v); Added in API level 24

half3 tanpi(half3 v); Added in API level 24

half4 tanpi(half4 v); Added in API level 24

Returns the tangent of (v * pi), where (v * pi) is measured in radians.

To get the tangent of a value measured in degrees, call `tanpi(v / 180.f)`.

See also native_tanpi().

tgamma : Gamma function

float tgamma(float v);

float2 tgamma(float2 v);

float3 tgamma(float3 v);

float4 tgamma(float4 v);

half tgamma(half v); Added in API level 24

half2 tgamma(half2 v); Added in API level 24

half3 tgamma(half3 v); Added in API level 24

half4 tgamma(half4 v); Added in API level 24

Returns the gamma function of a value.

See also lgamma().

trunc : Truncates a floating point

float trunc(float v);

float2 trunc(float2 v);

float3 trunc(float3 v);

float4 trunc(float4 v);

half trunc(half v); Added in API level 24

half2 trunc(half2 v); Added in API level 24

half3 trunc(half3 v); Added in API level 24

half4 trunc(half4 v); Added in API level 24

Rounds to integral using truncation.

For example, `trunc(1.7f)` returns 1.f and `trunc(-1.7f)` returns -1.f.

See rint() and round() for other rounding options.