Cheatography

RenderScript Math Functions Cheat Sheet

by Alberto Marchetti (cmaster11) via cheatography.com/26309/cs/7330/

Intrinsic math functions

RenderScript provides a lot of intrinsic math functions, which are optimized using NEON instructions (on ARMv7 devices). These functions should be used instead of writing pure calculations when possible.
For example, `a * b + c` should be translated into `fma(a, b, c)`.

Note: functions' arguments are assumed to be of type that matches function's return one. Otherwise, different type is specified.

Following functions have been taken directly from Android source code.

Float functions (part 1)

Definition	Description
`acos(v)`	Inverse cosine
`acosh(v)`	Inverse hyperbolic cosine
`acospi(v)`	Inverse cosine divided by pi
`asin(v)`	Inverse sine
`asinh(v)`	Inverse hyperbolic sine
`asinpi(v)`	Inverse sine divided by pi
`atan(v)`	Inverse tangent
`atan2(num, den)`	Inverse tangent of a ratio
`atan2pi(num, den)`	Inverse tangent of a ratio, divided by pi
`atanh(v)`	Inverse hyperbolic tangent
`atanpi(v)`	Inverse tangent divided by pi
`cbrt(v)`	Cube root
`ceil(v)`	Smallest integer not less than a value

Float functions (part 1) (cont)

`clamp(value, min, max)`	Restrain a value to a range
`copysign(magnitude, sign)`	Copies the sign of a number to another. `copysign(4.0f, -2.7f)` returns `-4.0f`
`cos(v)`	Cosine
`cosh(v)`	Hypebolic cosine
`cospi(v)`	Cosine of a number multiplied by pi
`degrees(v)`	Converts radians into degrees
`erf(v)`	Mathematical error function
`erfc(v)`	Mathematical complementary error function
`exp(v)`	*e* raised to a number
`exp10(v)`	10 raised to a number
`exp2(v)`	2 raised to a number
`expm1(v)`	*e* raised to a number minus one: `(e ^ v) - 1`
`fabs(v)`	Absolute value of a float
`fdim(a, b)`	Positive difference between two values. If `a > b`, returns `(a - b)` otherwise returns `0f`
`floor(v)`	Smallest integer not greater than a value/ `(multiplicand1 * multiplicand2) + offset`
`fma(mul1, mul2, offset)`	Multiply and add
`fmax(a, b)`	Maximum of two floats
`fmin(a, b)`	Minimum of two floats

Float functions (part 1) (cont)

`fmod(num, den)`	Modulo
`fract(v, float* floor)`	Positive fractional part. `fract(2.3f, &val)` returns `0.3f` and sets val to `2.f`
`frexp(v, int* exponent)`	Binary mantissa and exponent
`half_recip(v)`	Reciprocal computed to 16 bit precision
`half_rsqrt(v)`	Reciprocal of a square root computed to 16 bit precision
`half_sqrt(v)`	Square root computed to 16 bit precision

Predefined constants

Type	Name	Value
`float`	`M_PI`	`3.141592... (pi)`
`float`	`M_PI_2`	`M_PI / 2`
`float`	`M_PI_4`	`M_PI / 4`
`float`	`M_1_PI`	`1 / M_PI`
`float`	`M_2_PI`	`2 / M_PI`
`float`	`M_2_SQRTPI`	`2 / sqrt(M_PI)`
`float`	`M_SQRT2`	`sqrt(2)`
`float`	`M_E`	`2.718281... (e)`
`float`	`M_LN10`	`log_e(10)`
`float`	`M_LN2`	`log_e(2)`
`float`	`M_LOG10E`	`log_10(M_E)`
`float`	`M_LOG2E`	`log_2(M_E)`
`float`	`M_SQRT1_2`	`1 / sqrt(2)`

Float functions (part 2)

Definition	Description
`hypot(a, b)`	Hypotenuse

By **Alberto Marchetti** (cmaster11)
cheatography.com/cmaster11/
hydex11.net

Not published yet.
Last updated 25th February, 2016.

Float functions (part 2) (cont)

`ldexp(mantissa, int exponent)`	Creates a floating point from mantissa and exponent. `mantissa * 2 ^ exponent`
`lgamma(v)`	Natural logarithm of the gamma function
`lgamma(v, int* sign_of_gamma)`	Natural logarithm of the gamma function. 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`
`log(v)`	Natural logarithm
`log10(v)`	Base 10 logarithm
`log1p(v)`	Natural logarithm of a value plus 1
`log2(v)`	Base 2 logarithm
`logb(v)`	Base two exponent. `logb(8.5f)` returns `3.f`
`mad(mul1, mul2, offset)`	Multiply and add
`max(a, b)`	Maximum
`min(a, b)`	Minimum
`modf(v, float* integral_part)`	Integral and fractional components. `modf(-3.72f)` will return `0.72f` and `integral_part` will be set to `-3.f`
`nan(uint v)`	Returns NaN
`nextafter(v, target)`	Next representable floating point number from v towards target
`pow(base, exp)`	Base raised to an exponent

Float functions (part 2) (cont)

`pown(base, int expt)`	Base raised to an integer exponent
`powr(base, exp)`	Positive base raised to an exponent
`radians(v)`	Converts degrees into radians
`remainder(num, den)`	Remainder of a division
`remquo(num, int* quotient)`	Remainder and quotient of a division
`rint(v)`	Round to even
`rootn(v, int n)`	Nth root
`round(v)`	Round away from zero
`rsqrt(v)`	Reciprocal of a square root
`sign(v)`	Sign of a value
`sin(v)`	Sine
`sincos(v, float* cos)`	Sine and cosine
`sinh(v)`	Hyperbolic sine
`sinpi(v)`	Sine of a number multiplied by pi
`sqrt(v)`	Square root
`step(edge, v)`	Returns `0.f` if `v < edge`, `1.f` otherwise
`tan(v)`	Tangent
`tanh(v)`	Hyperbolic tangent
`tanpi(v)`	Tangent of a number multiplied by pi
`tgamma(v)`	Gamma function
`trunc(v)`	Truncates a floating point

Integer functions (return int)

Definition	Description
`abs(v)`	Absolute value of an integer
`clamp(value, min, max)`	Restrain a value to a range (*min API 19*)
`clz(value)`	Number of leading 0 bits
`ilogb(float v)`	Base two exponent
`max(a, b)`	Maximum value of two arguments
`min(a, b)`	Minimum value of two arguments
`rsRand(max)`	Pseudo-random number
`rsRand(min, max)`	Pseudo-random number

Approximate float functions (API >= 21)

Following functions have stricter limits than precise ones. Please refer to specs to see them.

Definition	Description
`native_log2(v)`	Approximate base 2 logarithm (API 18)
`native_powr(base, exp)`	Approximate positive base raised to an exponent (API 18)
`native_acos(v)`	Approximate inverse cosine
`native_acosh(v)`	Approximate inverse hyperbolic cosine
`native_acospi(v)`	Approximate inverse cosine divided by pi
`native_asin(v)`	Approximate inverse sine
`native_asinh(v)`	Approximate inverse hyperbolic sine
`native_asinpi(v)`	Approximate inverse sine divided by pi

By **Alberto Marchetti** (cmaster11)
cheatography.com/cmaster11/
hydex11.net

Not published yet.
Last updated 25th February, 2016.

Approximate float functions (API >= 21) (cont)	
native_atan(v)	Approximate inverse tangent
native_atan2(num, den)	Approximate inverse tangent of a ratio
native_atan2pi(num, den)	Approximate inverse tangent of a ratio, divided by pi
native_atanh(v)	Approximate inverse hyperbolic tangent
native_atanpi(v)	Approximate inverse tangent divided by pi
native_cbrt(v)	Approximate cube root
native_cos(v)	Approximate cosine
native_cosh(v)	Approximate hypebolic cosine
native_cospi(v)	Approximate cosine of a number multiplied by pi
native_divide(left, right)	Approximate division
native_exp(v)	Approximate e raised to a number
native_exp10(v)	Approximate 10 raised to a number
native_exp2(v)	Approximate 2 raised to a number
native_expm1(v)	Approximate e raised to a number minus one
native_hypot(a, b)	Approximate hypotenuse
native_log(v)	Approximate natural logarithm
native_log10(v)	Approximate base 10 logarithm

Approximate float functions (API >= 21) (cont)	
native_log1p(v)	Approximate natural logarithm of a value plus 1
native_recip(v)	Approximate reciprocal
native_rootn(v, int n)	Approximate nth root
native_rsqrt(v)	Approximate reciprocal of a square root
native_sin(v)	Approximate sine
native_sincos(v, float* cos);	Approximate sine and cosine
native_sinh(v)	Approximate hyperbolic sine
native_sinpi(v)	Approximate sine of a number multiplied by pi
native_sqrt(v)	Approximate square root
native_tan(v)	Approximate tangent
native_tanh(v)	Approximate hyperbolic tangent
native_tanpi(v)	Approximate tangent of a number multiplied by pi

By **Alberto Marchetti**
(cmaster11)
cheatography.com/cmaster11/
hydex11.net

Not published yet.
Last updated 25th February, 2016.