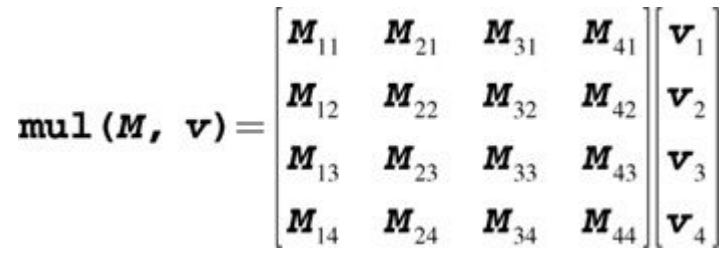
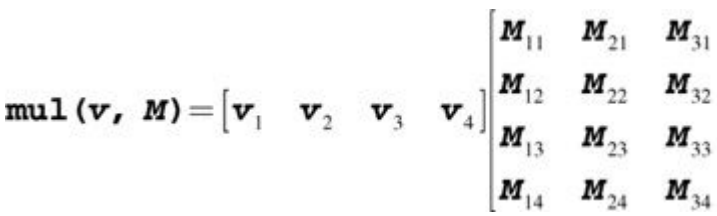
Function	Description
abs(x)	Absolute value of x .
acos(x)	Arccosine of x in range [0, π], x in [–1, 1].
all(x)	Returns true if every component of x is not equal to 0. Returns false otherwise.
any(x)	Returns true if any component of x is not equal to 0. Returns false otherwise.
asin(x)	Arcsine of x in range [–π/2, π/2]; x should be in [–1, 1].
atan(x)	Arctangent of x in range [–π/2, π/2].
atan2(y , x)	Arctangent of y **/** x in range [–π, π].
ceil(x)	Smallest integer not less than x .
clamp(x , a , b)	x clamped to the range [a , b] as follows: • Returns a if x is less than a . • Returns b if x is greater than b . • Returns x otherwise.
cos(x)	Cosine of x .
cosh(x)	Hyperbolic cosine of x .
cross(A , B)	Cross product of vectors A and B ; A and B must be three-component vectors.
degrees(x)	Radian-to-degree conversion.
determinant(M)	Determinant of matrix M .
dot(A , B)	Dot product of vectors A and B .
exp(x)	Exponential function e^x .
exp2(x)	Exponential function 2^x .
floor(x)	Largest integer not greater than x .
fmod(x , y)	Remainder of x **/** y , with the same sign as x . If y is 0, the result is implementation-defined.
frac(x)	Fractional part of x .
frexp(x , out exp)	Splits x into a normalized fraction in the interval [½, 1), which is returned, and a power of 2, which is stored in exp . If x is 0, both parts of the result are 0.
isfinite(x)	Returns true if x is finite.
isinf(x)	Returns true if x is infinite.
isnan(x)	Returns true if x is NaN (Not a Number).

Function	Description
ldexp(x , n)	$x \times 2^n$.
lerp(a , b , f)	Linear interpolation: (1 – f)* a + b * f where a and b are matching vector or scalar types. f can be either a scalar or a vector of the same type as a and b .
lit(NdotL , NdotH , m)	Computes lighting coefficients for ambient, diffuse, and specular light contributions. Expects the NdotL parameter to contain N • L and the NdotH parameter to contain N • H . Returns a four-component vector as follows: • The x component of the result vector contains the ambient coefficient, which is always 1.0. • The y component contains the diffuse coefficient, which is 0 if (N • L) < 0; otherwise (N • L). • The z component contains the specular coefficient, which is 0 if either (N • L) < 0 or (N • H) < 0; $(N \bullet H)^m$ otherwise. • The w component is 1.0. There is no vectorized version of this function.
log(x)	Natural logarithm ln(x) ; x must be greater than 0.
log2(x)	Base 2 logarithm of x ; x must be greater than 0.
log10(x)	Base 10 logarithm of x ; x must be greater than 0.
max(a , b)	Maximum of a and b .
min(a , b)	Minimum of a and b .
modf(x , out ip)	Splits x into integral and fractional parts, each with the same sign as x . Stores the integral part in ip and returns the fractional part.
mul(M , N)	Matrix product of matrix M and matrix N , as shown below: $\mathbf{mul}(M, N) = \begin{bmatrix} M_{11} & M_{21} & M_{31} & M_{41} \\ M_{12} & M_{22} & M_{32} & M_{42} \\ M_{13} & M_{23} & M_{33} & M_{43} \\ M_{14} & M_{24} & M_{34} & M_{44} \end{bmatrix} \begin{bmatrix} N_{11} & N_{21} & N_{31} \\ N_{12} & N_{22} & N_{32} \\ N_{13} & N_{23} & N_{33} \\ N_{14} & N_{24} & N_{34} \end{bmatrix}$

Function	Description
	If M has size A x B , and N has size B x C , returns a matrix of size A x C .
mul(M , v)	Product of matrix M and column vector v , as shown below:  If M is an A x B matrix and v is a B x 1 vector, returns an A x 1 vector.
mul(v , M)	Product of row vector v and matrix M , as shown below:  If v is a 1 x A vector and M is an A x B matrix, returns a 1 x B vector.
noise(x)	Either a one-, two-, or three-dimensional noise function, depending on the type of its argument. The returned value is between 0 and 1, and is always the same for a given input value.
pow(x , y)	x^y .
radians(x)	Degree-to-radian conversion.
round(x)	Closest integer to x .
rsqrt(x)	Reciprocal square root of x ; x must be greater than 0.
saturate(x)	Clamps x to the [0, 1] range.
sign(x)	1 if x > 0; –1 if x < 0; 0 otherwise.
sin(x)	Sine of x .
sincos(float x , out s , out c)	s is set to the sine of x , and c is set to the cosine of x . If both sin(x) and cos(x) are needed, this function is more efficient than calculating each individually.
sinh(x)	Hyperbolic sine of x .
smoothstep(min , max , x)	For values of x between min and max , returns a smoothly varying value that ranges from 0 at x = min to 1

Function	Description
	at x = max . x is clamped to the range [min , max] and then the interpolation formula is evaluated: $-2*((x - min)/(max - min))^3 +$ $3*((x - min)/(max - min))^2$
step(a , x)	0 if x < a ; 1 if x >= a .
sqrt(x)	Square root of x ; x must be greater than 0.
tan(x)	Tangent of x .
tanh(x)	Hyperbolic tangent of x .
transpose(M)	Matrix transpose of matrix M . If M is an A x B matrix, the transpose of M is a B x A matrix whose first column is the first row of M , whose second column is the second row of M , whose third column is the third row of M , and so on.
Function	**Description**
tex1D(sampler1D tex , float s)	1D nonprojective texture query
tex1D(sampler1D tex , float s , float dsdx , float dsdy)	1D nonprojective texture query with derivatives
tex1D(sampler1D tex , float2 sz)	1D nonprojective depth compare texture query
tex1D(sampler1D tex , float2 sz , float dsdx , float dsdy)	1D nonprojective depth compare texture query with derivatives
tex1Dproj(sampler1D tex , float2 sq)	1D projective texture query
tex1Dproj(sampler1D tex , float3 szq)	1D projective depth compare texture query
tex2D(sampler2D tex , float2 s)	2D nonprojective texture query
tex2D(sampler2D tex , float2 s , float2 dsdx ,	2D nonprojective texture query with derivatives

Function	Description
float2 dsdy)	
tex2D(sampler2D tex , float3 sz)	2D nonprojective depth compare texture query
tex2D(sampler2D tex , float3 sz , float2 dsdx , float2 dsdy)	2D nonprojective depth compare texture query with derivatives
tex2Dproj(sampler2D tex , float3 sq)	2D projective texture query
tex2Dproj(sampler2D tex , float4 szq)	2D projective depth compare texture query
texRECT(samplerRECT tex , float2 s)	2D nonprojective texture rectangle texture query (OpenGL only)
texRECT(samplerRECT tex , float2 s , float2 dsdx , float2 dsdy)	2D nonprojective texture rectangle texture query with derivatives (OpenGL only)
texRECT(samplerRECT tex , float3 sz)	2D nonprojective texture rectangle depth compare texture query (OpenGL only)
texRECT(samplerRECT tex , float3 sz , float2 dsdx , float2 dsdy)	2D nonprojective depth compare texture query with derivatives (OpenGL only)
texRECTproj(samplerRECT tex , float3 sq)	2D texture rectangle projective texture query (OpenGL only)
texRECTproj(samplerRECT tex , float3 szq)	2D texture rectangle projective depth compare texture query (OpenGL only)
tex3D(sampler3D tex , float3 s)	3D nonprojective texture query
tex3D(sampler3D tex , float3 s , float3 dsdx , float3 dsdy)	3D nonprojective texture query with derivatives
tex3Dproj(sampler3D tex , float4 sq)	3D projective texture query
texCUBE(samplerCUBE tex , float3 s)	Cube map nonprojective texture query
texCUBE(samplerCUBE tex , float3 s , float3 dsdx , float3 dsdy)	Cube map nonprojective texture query with derivatives
texCUBEproj(samplerCUBE tex , float4 sq)	Cube map projective texture query (ignores q)