libfpu Reference

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Table of Contents

FPU Basic Operation Functions

sceFpuAbsf

Absolute value

Definition

```
#include <libfpu.h>
float sceFpuAbsf(
        float s
);
```

Calling Conditions

Multithread safe

Arguments

s	Floating-point value

Return Value

The absolute value of *s* is returned.

Description

This function calculates the absolute value.

See Also

`fabsf()`

Trigonometric Functions

sceFpuSinf

Sine

Definition

```
#include <libfpu.h>
float sceFpuSinf(
        float s
);
```

Calling Conditions

Multithread safe

Arguments

s Floating-point value

Return Value

`sin(s)` is returned.

Description

This function performs polynomial expansion in FPU to calculate `sin(s)`.

If the absolute value of *s* is π or less, the error compared to the value of `sin` that was obtained precisely will be 1 LSB or less.

If the absolute value of *s* exceeds about $2^{31}\pi$, the correct value will not be returned. In addition, there is a possibility that a FPU exception occurs.

See Also

`sinf()`

sceFpuCosf

Cosine

Definition

```
#include <libfpu.h>
float sceFpuCosf(
        float s
);
```

Calling Conditions

Multithread safe

Arguments

s	Floating-point value

Return Value

`cos(s)` is returned.

Description

This function performs polynomial expansion in FPU to calculate `cos(s)`.

If the absolute value of *s* is π or less, the error compared to the value of `cos` that was obtained precisely will be 1 LSB or less.

If the absolute value of *s* exceeds about $2^{31}\pi$, the correct value will not be returned. In addition, there is a possibility that a FPU exception occurs.

See Also

`cosf()`

sceFpuTanf

Tangent

Definition

```
#include <libfpu.h>
float sceFpuTanf(
        float s
);
```

Calling Conditions

Multithread safe

Arguments

s	Floating-point value

Return Value

`tan(s)` is returned.

Description

This function uses the FPU to calculate the tangent of *s* according to a polynomial expansion.

If the absolute value of *s* is π or less, the error relative to the exact value of `tan` will be 1 LSB or less.

If the absolute value of *s* exceeds about $2^{31}\pi$, the correct value will not be returned. In addition, there is a possibility that a FPU exception occurs.

See Also

`tanf()`

sceFpuAcosf

Arc cosine

Definition

```
#include <libfpu.h>
float sceFpuAcosf(
        float s
);
```

Calling Conditions

Multithread safe

Arguments

s Floating-point value

Return Value

`acos(s)` is returned.

Description

This function uses the FPU to calculate the arc cosine of *s* according to a polynomial expansion. Regardless of the range of *s*, the error relative to the exact value of `acos` will be 1 LSB or less.

See Also

`acosf()`

sceFpuAsinf

Arc sine

Definition

```
#include <libfpu.h>
float sceFpuAsinf(
        float s
);
```

Calling Conditions

Multithread safe

Arguments

s Floating-point value

Return Value

`asin(s)` is returned.

Description

This function uses the FPU to calculate the arc sine of *s* according to a polynomial expansion. Regardless of the range of *s*, the error relative to the exact value of `asin` will be 1 LSB or less.

See Also

`asinf()`

sceFpuAtanf

Arc tangent

Definition

```
#include <libfpu.h>
float sceFpuAtanf(
        float s
);
```

Calling Conditions

Multithread safe

Arguments

s	Floating-point number

Return Value

`atan(s)` is returned.

Description

This function uses the FPU to calculate the arc tangent of `s` according to a polynomial expansion. Regardless of the range of *s*, the error relative to the exact value of `atan` will not exceed 1 LSB.

See Also

`atanf()`

sceFpuAtan2f

Arc tangent

Definition

```
#include <libfpu.h>
float sceFpuAtan2f(
        float x,
        float y
);
```

Calling Conditions

Multithread safe

Arguments

x	Floating-point value
y	Floating-point value

Return Value

`atan(x/y)` is returned.

Description

This function uses the FPU to calculate the arc tangent of *x*/*y* according to a polynomial expansion. Regardless of the ranges of *x* and *y*, the error relative to the exact value of `atan2` will be 1 LSB or less.

See Also

`atan2f()`

Logarithmic/Exponential Functions

sceFpuLogf

Natural Logarithm

Definition

```
#include <libfpu.h>
float sceFpuLogf(
        float s
);
```

Calling Conditions

Multithread safe

Arguments

s Floating-point value

Return Value

`log(s)` is returned.

Description

This function performs polynomial expansion in FPU to calculate log(*s*).

The error compared to the value of `log` that was obtained precisely will be 2 LSB or less.

See Also

`logf()`

sceFpuLog2f

Logarithm base 2

Definition

```
#include <libfpu.h>
float sceFpuLog2f(
        float s
);
```

Calling Conditions

Multithread safe

Arguments

s Floating-point value

Return Value

`log2(s)` is returned.

Description

This function uses the FPU to calculate the base 2 logarithm of *s* using a polynomial expansion.

The error when compared with the precise value of `log2` is less than or equal to 2 LSB.

See Also

`log2f()`

sceFpuLog10f

Common logarithm

Definition

```
#include <libfpu.h>
float sceFpuLog10f(
        float s
);
```

Calling Conditions

Multithread safe

Arguments

s	Floating-point value

Return Value

`log10(s)` is returned.

Description

This function uses the FPU to calculate the common logarithm of *s* using a polynomial expansion.

The error when compared with the precise value of `log10` is less than or equal to 2 LSB.

See Also

`log10f()`

sceFpuExpf

Exponential base e

Definition

```
#include <libfpu.h>
float sceFpuExpf(
        float x
);
```

Calling Conditions

Multithread safe

Arguments

x	Floating-point value

Return Value

e^{x} is returned.

Description

This function uses the FPU to calculate `e` to the *x* power using a polynomial expansion.

The error compared to the value of `exp` that was obtained precisely will be 3 LSB or less.

See Also

`expf()`

sceFpuExp2f

Exponential base 2

Definition

```
#include <libfpu.h>
float sceFpuExp2f(
        float x
);
```

Calling Conditions

Multithread safe

Arguments

x Floating-point value

Return Value

2^x is returned.

Description

This function uses the FPU to calculate 2 to the *x* power using a polynomial expansion.

The error compared to the value of 2^x that was obtained precisely will be 3 LSB or less.

See Also

```
exp2f()
```

sceFpuExp10f

Exponential base 10

Definition

```
#include <libfpu.h>
float sceFpuExp10f(
        float x
);
```

Calling Conditions

Multithread safe

Arguments

x	Floating-point value

Return Value

10^x is returned.

Description

This function uses the FPU to calculate 10 to the *x* power using a polynomial expansion.

The error compared to the value of 10^x that was obtained precisely will be 3 LSB or less.

See Also

`exp10f()`

sceFpuPowf

Power function

Definition

```
#include <libfpu.h>
float sceFpuPowf(
        float x,
        float y
);
```

Calling Conditions

Multithread safe

Arguments

x	Floating-point value
y	Floating-point value

Return Value

x^y is returned.

When *x* is negative and *y* is not an integer value, the return value is NaN.

Description

This function uses the FPU to calculate *x* to the *y* power using a polynomial expansion.

When *x* is negative and *y* is not an integer value, a NaN is generated. To determine whether or not *y* is an integer value, cast *y* to integer type and convert the result into `float` type again, and then see if the calculation result is equal to *y*. Be careful when the absolute value of *y* exceeds the precision of a `float`.

See Also

`powf()`