User's Manual

NEC

GOFAST

Floating Point Library

Ver 2.0

Target Devices

V850 Series™

Document No. SUD-DT-03-0126-E (3rd edition)

Date Published March 2003 N CP(K)

U S Software, A Lantronix Company 7175 NW Evergreen Parkway #100 Hillsboro, OR 97124

- V850 Series is trademarks of NEC Electronics Corporation.
- Green Hills Software and MULTI are trademarks of Green Hills Software, Inc.
- Wind River, Tornado and VxWorks are trademarks or registered trademarks of Wind River Systems, Inc.
- GNUPro and red hat are trademarks or registered trademark of Red Hat, Inc.
- GOFAST is trademark of Lantronix, Inc.
- IAR Embedded Workbench is trademark of IAR Systems.

and/or other countries. All other marks or trademarks are property of their respective holders.

The export of these products from Japan is regulated by the Japanese government. The export of some or all of these products may be prohibited without governmental license. To export or re-export some or all of these products from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

Major Revisions in This Edition

Ver.	Issued Date	Description
01	2001.12.19	(1st edition) V850 Series GOFAST Floating Point Library User's Manual
02	2002.11.15	Addtion of GHS –pid and –registermode=22
03	2003.03.18	Addtion of IAR EWV850

INTRODUCTION

Target:

The V850 Series GOFAST Floating Point Library is a floating-point library created by using Lantronix's FPT 3.0.

This library is designed for the following.

- V850/V850E for RedHat GNU Pro, for GHS and for IAR compiler

Target Version:

The following version is targeted in this manual.

V850 Series GOFAST Floating Point Library Ver. 2.0 or later

Purpose:

This manual explains the functions provided in the V850 Series GOFAST Floating Point Library.

CONTENTS

1. OVERVIEW

This section explains the features of the V850 Series GOFAST Floating Point Library.

1.1. Features of V850 Series GOFAST Floating Point Library

The V850 Series GOFAST Floating Point Library is a floating-point library created by using USSOFT FPT 3.0.

This library provides arithmatic functions comforming to the ANSI C (JIS X 3010) Standard.

Fundamental operations

Double precision:	dpadd, dpsub, dpmul, dpdiv
Single precision:	fpadd, fpsub, fpmul, fpdiv
Conversion function:	dptofp, litodp, dptoli, ultodp, dptoul, litofp, fptoli, ultofp, fptoul GCC GHS only: lltodp, ulltodp, dptoull, dptoll, fptoll, fptoull, lltofp, ulltofp
Comparison:	dpcmp, fpcmp GCC only: __negdf2, __negsf2, ___eqdf2, ___nedf2, ___ltdf2, ___ledf2, ___gtdf2, ___gedf2, ___eqsf2, ___nesf2, ___ltsf2, ___lesf2, ___gtsf2, ___gesf2

Fundamental functions

Double precision:	fabs, ceil, floor, fmod, modf, frexp, ldexp, sqrt
Single precision:	fabsf, ceilf, floorf, fmodf, modff, frexpf, ldexpf, sqrtf

Transcendental functions

Double precision:	asin, acos, atan, atan2, cos, cosh, exp, log, log10, pow, sin, sinh, tan, tanh
Single precision:	acosf, asinf, atanf, atan2f, cosf, coshf, expf, logf, log10f, powf, sinf, sinhf, tanf, tanhf

Additional functions isnan, isnanf, isinf, isinff

2. OPERATING ENVIRONMENT

2.1. Operating Environment

The V850 Series GOFAST Floating Point Library can be used in the following compilers.

Target Compiler	Compiler Whose Operation Has Been Confirmed
GHS compiler for V850/V850E	MULTI 1.8.9, MULTI 2000
GNU Pro compiler for V850/V850E	GNUPro v850ice-011005 for NEC V850
GNU Pro compiler for V850/V850E (longcall)	Wind River Systems Tornado 2.0 for NEC V850E
IAR compiler for EWV850	IAR Embedded Workbench NEC V850/V850E Series EWV850 Ver. 2.11A

2.2. File Configuration

This section explains the file configuration included in the V850 Series GOFAST Floating Point Library.

The directory configuration of the supplied files is shown below.

```
GOFAST_V850
        examples
        include
        v850-ghs v850-ghs-pid v850-ghs-reg22 v850-ghs-reg22-pid
        v850e-ghs v850e-ghs-pid v850e-ghs-reg22 v850e-ghs-reg22-pid
        v850-cyg v850-cyg-longcall
        v850e-cyg v850e-cyg-longcall
        v850-iar-ll v850-iar-ln v850-iar-lp
        v850-iar-sl v850-iar-sn v850-iar-sp
        v850-iar-tl v850-iar-tn
        v850e-iar-ll v850e-iar-ln v850e-iar-lp
        v850e-iar-sl v850e-iar-sn v850e-iar-sp
        v850e-iar-tl v850e-iar-tn
doc
        GOFAST_V850
Util
```

Two kinds of files, files dependent on the target and compiler, and common files, are provided in the V850 Series GOFAST Floating Point Library. Files dependent on the target and compiler are included in the corresponding library directory shown below. Read <library directory> in this document as the "corresponding library directory".

Compiler		Library Directory
GNUPro/Tornado	V850	v850-cyg
	V850E	v850e-cyg
	V850 (for longcall)	v850-cyg-longcall
	V850E (for longcall)	v850e-cyg-longcall
GHS	V850	v850-ghs
	V850E	v850e-ghs
	V850 –pid	v850-ghs-pid
	V850E –pid	v850e-ghs-pid
	V850 –registermode=22	v850-ghs-reg22
	V850E –registermode=22	v850e-ghs-reg22
	V850 –registermode=22 –pid	v850-ghs-reg22-pid
	V850E –registermode=22 –pid	v850e-ghs-reg22-pid
IAR EWV850	V850 -ms -code_model normal	v850-iar-sn
	V850 -mt -code_model normal	v850-iar-tn
	V850 -ml -code_model normal	v850-iar-ln
	V850 -ms -code_model large	v850-iar-sl
	V850 -mt -code_model large	v850-iar-tl
	V850 -ml -code_model large	v850-iar-ll
	V850 -ms -code_model pic	v850-iar-sp
	V850 -ml -code_model pic	v850-iar-lp
	V850e -ms -code_model normal	v850e-iar-sn
	V850e -mt -code_model normal	v850e-iar-tn
	V850e -ml -code_model normal	v850e-iar-ln
	V850e -ms -code_model large	v850e-iar-sl
	V850e -mt -code_model large	v850e-iar-tl
	V850e -ml -code_model large	v850e-iar-ll
	V850e -ms -code_model pic	v850e-iar-sp
	V850e -ml -code_model pic	v850e-iar-lp

The files in each directory are shown below.

Document Directory: doc/GOFAST_V850
PDF File

Header file Directory: GOFAST_V850/include
File: math.h, float.h

Samples Directory: GOFAST_V850/examples

File: whetstone.c (Available from NetLib http://www.netlib.org/)

Library and source file Directory: GOFAST_V850/<library directory>

Mathematical function library file: **libgofast.a**

A symbol list file that converts the mathematical function symbols included in the GHS compiler into local symbols (GHS compiler only): **retainsymbollist.sample**

A make file: **Makefile**

The following files are also included in the source version.

Assembler source file (*.s)

make file: **Makefile**

Only in GHS compiler: MULTI 1.8.9 bld file **libgofast**, **multi189.bld**

MULTI 2000 bld file **libgofast**, **multi2000.bld**

Only in IAR compiler: bat file **Makefile.bat**

All the text files employ the newline code for Windows. Files employing the newline code for UNIX are provided as GOFAST_V850/lib_unix.tgz (binary version) or GOFAST_V850/src_unix.tgz (source version), in which the above files have been tar + gzip processed.

3. USAGE

3.1. Installation

This section explains the installation method of the V850 Series GOFAST Floating Point Library.
Copy the directory **include** that is included in the attachment and the library directory corresponding to the target and compiler to the installation directory.

3.2. Compilation

This section explains the compilation method using the V850 Series GOFAST Floating Point Library.

3.2.1. Specification method at compilation

Use the compiler as shown below.

In case of compiler for GNU Pro

It is not required to specify the compiler option **-lm**. Perform compilation as shown below.

<compiler for target> <compiler option> -I<installation directory>/include source file
-L<installation directory>/<library directory> -lc -lgofast

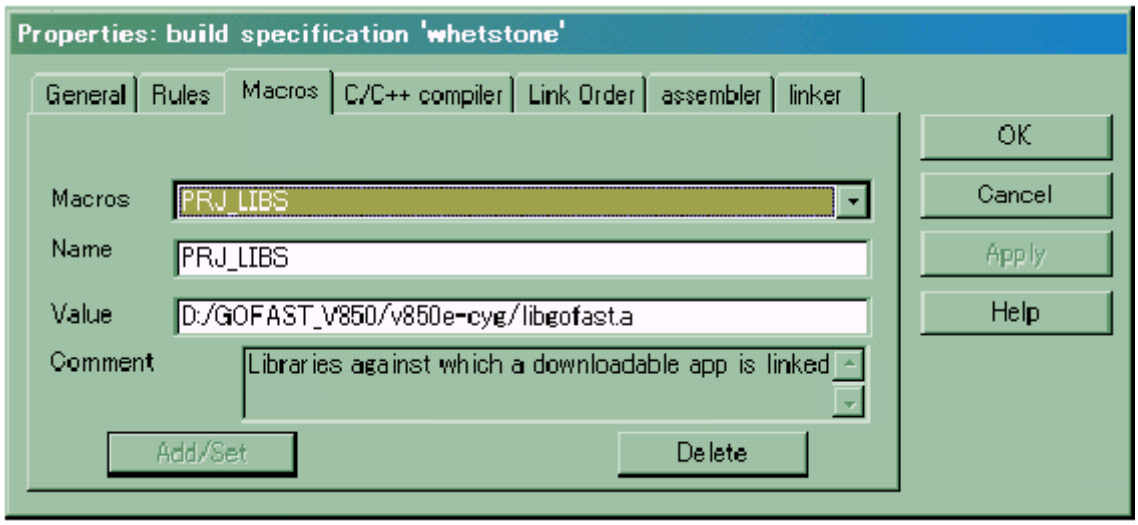
In case of compiler for Tornado

Perform compilation in the same manner as for the GNU Pro when compiling via the command line.

Specify the library from Tornado as shown below.

Creation of downloadable application

Specify the library by selecting PRJ_LIBS from the Macros tab in the Build Property Sheet window.

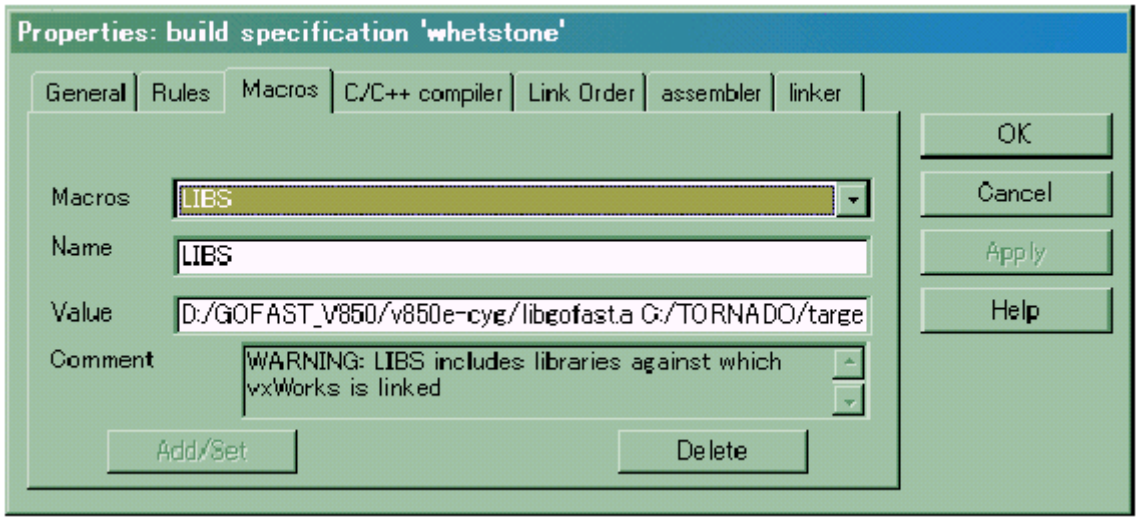


Set the library **libgofast.a** to the Value field as shown below.

D:/GOFAST_V850/v850e-cyg/libgofast.a

Creation of bootable application

Specify the library by selecting LIBS from the Macros tab in the Build Property Sheet window.



Set the library **libV850Egnuvx.a** to the Value field so that **libV850Egnuvx.a** is between two "**libgofast.a**"s as shown below.

D:/GOFAST_V850/v850e-cyg/libgofast.a
C:/TORNADO/target/lib/libV850Egnuvx.a
D:/GOFAST_V850/v850e-cyg/libgofast.a
C:/TORNADO/host/x86-win32/lib/gcc-lib/v850-wrs-vxworks/2.9-gnupro-99r1/libgcc.a

In case of GHS compiler

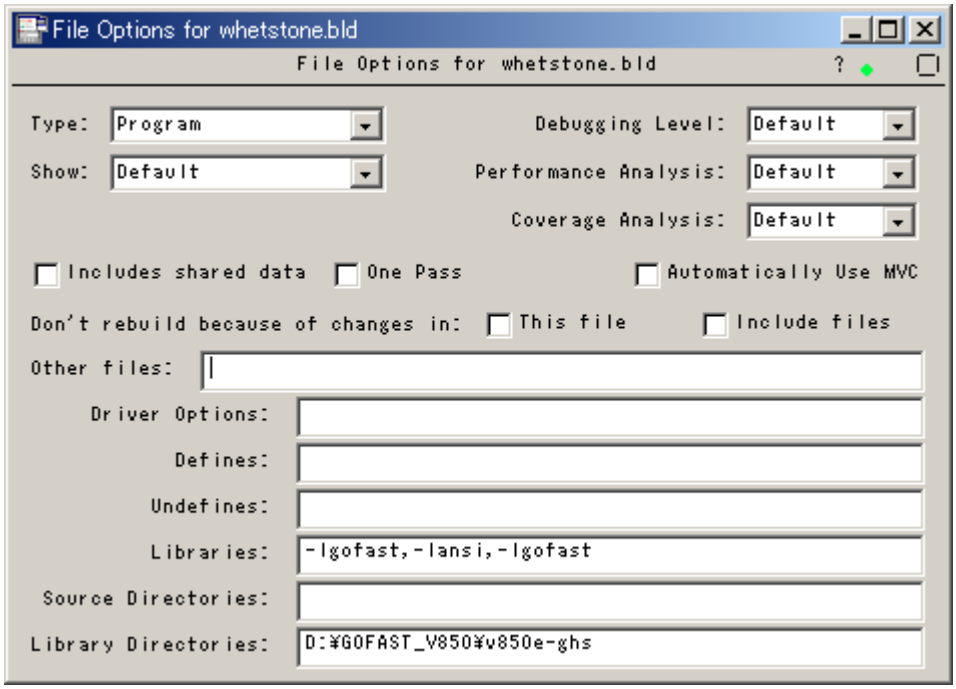
Perform compilation as shown below.

When the library **libansi.a** is used, specify **-lansi** so that it is between two **-lgofast**s. The use of **libansi.a** can be confirmed when linking the files by specifying **-v** to the compiler. In MULTI, select "Command" from the [Build] → [Build Panel] menu. In MULTI 2000, select "Commands" in the [DisplayOverides:] field under the [Build] → [Advanced Build Controls...] menu.

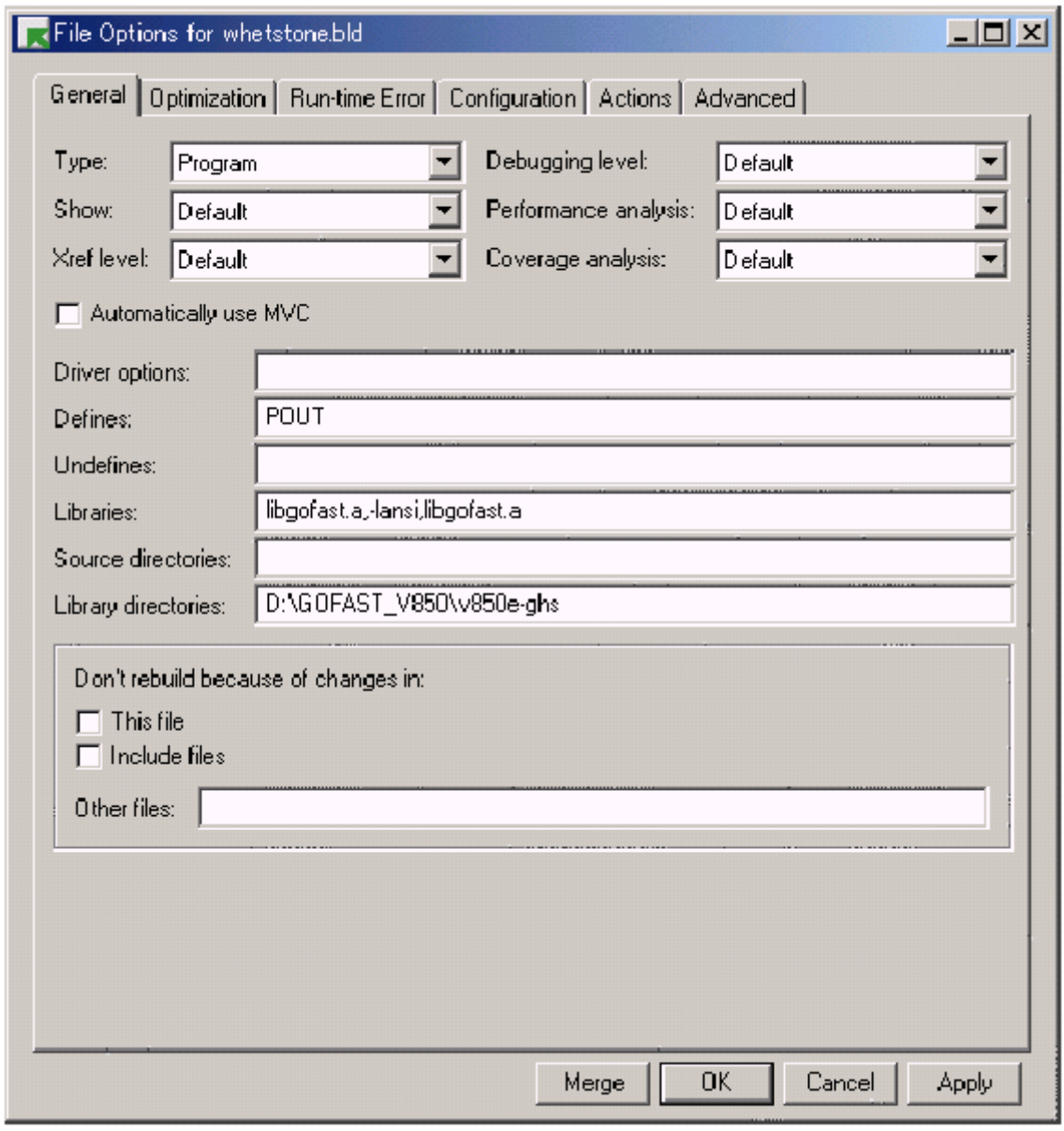
Perform compilation as shown below when compiling via the command line.

<compiler for target> <compiler option> -I<installation>/include source file -L<installation directory>/<library directory> -lgofast -lansi -lgofast

When using MULTI 1.8.9, specify the library in the Libraries: field by opening the window from the [Options] → [General Option] menu. A setting example using MULTI 1.8.9 is shown below.



When using MULTI 2000, specify the library in the Libraries: and “Library directories:” fields by opening the window from the [Project] → [File Options for...] menu. A setting example using MULTI 2000 is shown below.



In case of IAR compiler

Add libgofast.r85 to “Project Files” from Embedded Workbench IDE.

Perform compilation as shown below when compiling via the command line.

<linker xlink> <linker options 1> <user objects> <library libgofast.r85> <linker options 2> <runtime library cl*.r85> <linker options 3>

3.2.2. Supplement for specification method of libgofast.a

The GCC and GHS compilers have two libraries: a C language library and runtime library. The C language library includes functions such as **sin()** and **long(),** which are used for calling functions in C language. The runtime library includes functions used for addition, subtraction, multiplication, and division of the floating point decimal (FLDEC). When the program is compiled by the user, the linker links these libraries and creates object files.

For example, assume a situation in which there are calls for functions **c_func1()**, **c_func2()**, and **r_func1()** in a C source file, the entities of functions **c_func1()** and **c_func2()** exist in the C language library, and the entity of function **r _unc1()** exists in the runtime library. When linking these files, first link functions **c_func1()** and **c_func2()** from the C language library, and then link function **r_func1()** from the runtime library. At this time, if function **r_func2()** existing in the runtime library is called from function **c_func2()** existing in the C language library, the linker first links function **c_func2()** existing in the C language library, then links function **r_func2()** together with function **r_func1()** in the runtime library. An operation example at this time is shown in Figure 1. In Figure 1, the function referenced from the object file in the leftmost column is searched from the C language library and runtime library, and the function object found is output to the execution format file in the rightmost column.

Input			Output
Object File	C Language Library	Runtime Library	Execution Format File

c_func1();	c_func1()
	{
	}
		→	c_func1()
			{
			}

c_func2();	c_func2()
	{
	r_func2();
	}
		→	c_func2()
			{
			r_func2();
			}

		r_func2();
		{
		}
			r_func2();
			{
			}

r_func1();		r_func1();
		{
		}
			r_func1();
			{
			}

Figure 1: Link Image of Object File, C Language Library and Runtime Library

When using the library **libgofast.a** in the V850 Series GOFAST Floating Point Library, the linking is performed in the same manner. The library **libgofast.a** includes the functions that are in the C language library and runtime library. Therefore, more effective usage of GOFAST can be realized by specifying the following.

-lgofast -l<C language library> -lgofast

With this specification, the linker links files in the following order.

(a)	(b)	(c)	(d)
Library **libgofast.a** →	C language library →	Library **libgofast.a** →	Runtime library

The actual operation of the linker is shown below.

(1) The linker searches functions **c_func1()** and **c_func2()** from the library **libgofast.a** and links them. In addition, function **r_func2()** is added to the link target.
(2) Functions **c_func1()** and **c_func2()** in the C language library have already been linked from the library **libgofast.a** in (1). Therefore, they are not linked from the C language library.
(3) The linker searches function **r_func2()**, which is called from functions **r_func1()** and **c_func2()**, from the library **libgofast.a** again and links them.
(4) Functions **r_func1()** and **r_func2()** in the runtime library have already been linked from the library **libgofast.a** in (3). Therefore, they are not linked from the runtime library.

An example of creating the exe format by using the library **libgofast.a** is shown in Figure 2.

Input					Output
Object File	**libgofast.a**	C Language Library	libgofast.a	Runtime Library	Execution Format File

c_func1();	c_func1()
	{
	}
		→	→	→	c_func1()
					{
					}

		c_func1()
		{
		}			

			c_func1()
			{
			}		

c_func2();	c_func2()
	{
	r_func2();
	}
		→	→	→	c_func2()
					{
					r_func2();
					}

		c_func2()
		{
		r_func2();
		}			

			r_func2();
			{
			}
				→	r_func2();
					{
					}

				r_func2();
				{
				}	

r_func1();	r_func1()
	{
	}
		→	→	→	r_func1();
					{
					}

			r_func1();
			{
			}		

				r_func1();
				{
				}	

Figure 2: Link Image of Object File, libgofast.a, C Language Library, libgofast.a and Runtime Library

To confirm the reference relationships between the libraries, generally the "name (nm)" command[1] can be used. Refer to the user's manual of each compiler for details of usage.

[1] The command gnm in GHS, and the command v850-elf-nm (or equivalent) in GCC

3.3. Creation of Library from Included Library Source File

This section explains the library creation method using the library source file included in the compiler. The assembler source file, the object file **gofast.o** and the sample file **Makefile** are included in the library directory.

Change the following macros of **Makefile** according to the compiler used.

CC = C compiler/assembler command

CFLAGS = Options passed to C compiler/assembler (Set the macro so that the object file is created from the assembler file.)

AR = Archiver command

ARFLAGS = Archiver option (Set the macro so that the archive is created from an object file that has been passed.)

RANLIB = Library command

Create the library **libgofast.a** using the **make** command.

```
make libgofast.a
```

When creating the library **libgofast.a** by using the **bld** file in the GHS compiler, execute the following comannd via the MS-DOS prompt. In MULTI 2000, read libgofast.multi189.bld as libgofast.multi2000.bld.

```
build libgofast.multi189.bld
```

When creating the library **libgofast.a** by using the bat file Makefile.bat for IAR, execute the bat file with MS-DOS prompt.

3.4. Adjusting the Library

This section shows a workaround for problems caused by symbol resolution using the library **libgofast.a** and an existing library.

3.4.1. Changing the library included in GCC compiler to a weak symbol

Before changing the library, ensure that the library **libc.a** and **libgcc.a** included in the GCC compiler can be restored. An example of the library **libc.a** is shown below. For the library **libgcc.a**, implement the same operation by reading the library **libc.a** as **libgcc.a**.

(1) <objcopy for target> --weaken <libg.a included in GCC>

A example is shown below when changing specified symbols.

(1) <objcopy for target> --weaken-symbol <symbols> <libg.a included in GCC>

3.4.2. Integrating the library with the library included in GHS compiler

Using the command *make*:

Before changing the librayy, ensure that the library **libind.a** and **libansi.a** included in the GHS compiler can be restored.

When the library is created using the **make** command as shown below, the library **libind.a**, which is integrated with the V850 Series GOFAST Floating Point Library is created. Some mathematical functions are included in the library **libansi.a**. The library **libansi.a** created here is invalidated so that it does not overlap the mathematical functions included in the library **libind.a** that is integrated with the V850 Series GOFAST Floating Point Library.

Copy the created library **libind.a** and **libansi.a** to the GHS compiler library directory.

make libind.a GHS_ROOT=GHS compiler installation directory

Changing each library using comnand

Before changing the library, ensure that the library **libind.a** and **libansi.a** included in the GHS compiler can be restored. After that, implement the following procedure.

libind.a

(1) mkdir <work directory>
(2) cd <work directory>
(3) ax xv <libind.a included in the GHS compiler>
(4) ghide <symbol list file that will be retained in libind.a (e.g. file retain_symbollist.sample)> *.o
(5) ax xv libgofast.a
(6) ax crv libind.a *.o
(7) Copy the created library **libind.a** to the library **libind.a** included in the GHS compiler

libansi.a

(1) mkdir <work directory>
(2) cd <work directory>
(3) ax xv <libansi.a included in the GHS compiler>
(4) ghide <symbol list file that will be retained in libansi.a (e.g. file retain_symbollist.sample)> *.o
(5) ax crv libansi.a *.o
(6) Copy the created library **libansi.a** to the library **libansi.a** included in the GHS compiler

4. FLOATING-POINT LIBRARY

The V850 Series GOFAST Floating Point Library is a floating-point library conforming to the following standards.

- Floating-point representation system: IEEE 754
- Functions: ANSI C Standard (JIS X 3010)

4.1. Floating-point Type

The floating types that can be used in the V850 Series GOFAST Floating Point Library are shown below.

float	Single-precision floating-point type (32-bit)
double	Double-precision floating-point type (64-bit)

4.2. Functions

The functions that can be used in the V850 Series GOFAST Floating Point Library are explained below.

Function name:	A function name that can be used in C language. The usable compiler is described in parentheses but can be omitted if the function can be used in all the compilers.
Function type:	Description by function prototype in C language
Description type:	Description of the function
Return value:	Return value of the function
NaN:	Not a number. -NaN and +NaN are not distinguished.
+NaN:	NaN whose sign bit is 0
-NaN:	NaN whose sign bit is 1
Inf:	Infinity. Positive and negative are not distinguished.
+Inf:	Positive infinite number
-Inf:	Negative infinite number
DENORMAL:	Positive denormalized number. Positive and negative are not distinguished.
+DENORMAL:	Negative denormalized number
-DENORMAL:	Denormalized number
0:	0. Signed and unsigned are not distinguished.
+0:	0 whose sign bit is positive
-0:	0 whose sign bit is negative

4.2.1. Fundamental operations

Double precision

dpadd

Function name dpadd

Function type double dpadd (double x , double y)

Description Returns the result of adding x to y.

Return value

	y				
x	NaN	+INF	-INF	+0	-0
NaN	NaN	NaN	NaN	NaN	NaN
+INF	NaN	+INF	NaN	+INF	+INF
-INF	NaN	NaN	-INF	-INF	-INF
+0	NaN	+INF	-INF	0	0
-0	NaN	+INF	-INF	0	0

dpsub

Function name dpsub

Function type double dpsub (double x , double y)

Description Returns the result of subtracting y from x.

Return value

	y				
x	NaN	+INF	-INF	+0	-0
NaN	NaN	NaN	NaN	NaN	NaN
+INF	NaN	NaN	+INF	+INF	+INF
-INF	NaN	-INF	NaN	-INF	-INF
+0	NaN	-INF	+INF	0	0
-0	NaN	-INF	+INF	0	0

dpmul

Function name dpmul

Function type double dpmul (double x , double y)

Description Returns the result of mutiplying x and y.

Return value

	y				
x	NaN	+INF	-INF	+0	-0
NaN	NaN	NaN	NaN	NaN	NaN
+INF	NaN	+INF	-INF	NaN	NaN
-INF	NaN	-INF	+INF	NaN	NaN
+0	NaN	NaN	NaN	0	0
-0	NaN	NaN	NaN	0	0

dpdiv

Function name dpdiv

Function type double dpdiv (double x , double y)

Description Returns the result of dividing x by y.

Return value

	y				
x	NaN	+INF	-INF	+0	-0
NaN	NaN	NaN	NaN	NaN	NaN
+INF	NaN	NaN	NaN	+INF	-INF
-INF	NaN	NaN	NaN	-INF	+INF
+0	NaN	0	0	NaN	NaN
-0	NaN	0	0	NaN	NaN

Single precision

fpadd

Function name fpadd

Function type float fpadd (float x , float y)

Description Returns the result of adding x to y.

Return value

	y				
x	NaN	+INF	-INF	+0	-0
NaN	NaN	NaN	NaN	NaN	NaN
+INF	NaN	+INF	NaN	+INF	+INF
-INF	NaN	NaN	-INF	-INF	-INF
+0	NaN	INF	-INF	0	0
-0	NaN	INF	-INF	0	0

fpsub

Function name fpsub

Function type float fpsub (float x , float y)

Description Returns the result of subtracting y from x.

Return value

	y				
x	NaN	+INF	-INF	+0	-0
NaN	NaN	NaN	NaN	NaN	NaN
+INF	NaN	NaN	+INF	+INF	+INF
-INF	NaN	-INF	NaN	-INF	-INF
+0	NaN	-INF	+INF	0	0
-0	NaN	-INF	+INF	0	0

fpmul

Function name fpmul

Function type float fpmul (float x , float y)

Description Returns the result of multiplying x and y.

Return value

	y				
x	NaN	+INF	-INF	+0	-0
NaN	NaN	NaN	NaN	NaN	NaN
+INF	NaN	+INF	-INF	NaN	NaN
-INF	NaN	-INF	+INF	NaN	NaN
+0	NaN	NaN	NaN	0	0
-0	NaN	NaN	NaN	0	0

fpdiv

Function name fpdiv

Function type float fpdiv (float x , float y)

Description Returns the result of dividing x by y.

Return value

	y				
x	NaN	+INF	-INF	+0	-0
NaN	NaN	NaN	NaN	NaN	NaN
+INF	NaN	NaN	NaN	+INF	-INF
-INF	NaN	NaN	NaN	-INF	+INF
+0	NaN	0	0	NaN	NaN
-0	NaN	0	0	NaN	NaN

Conversion function

fptodp

Function name fptodp

Function type double fptodp (float x)

Description Returns the result of converting a single-precision floating-point number into a double-precision floating-point number.

Return value

x	fptodp (x)
x = NaN	NaN
x = +INF	+INF
x = -INF	-INF
x = 0	0

dptofp

Function name dptofp

Function type float dptofp (double x)

Description Returns the result of converting a double-precision floating-point number into a single-precision floating-point number.

Return value

x	dptofp(x)
$\|x\| >= 2^{128}$	Signed INF of x
$\|x\| <= 2^{-128}$	Signed 0 of x
x = NaN	NaN
x = +INF	+INF
x = -INF	-INF

litodp

Function name litodp

Function type double litodp (int x)

Description Returns the result of converting a signed integer into a double-precision floating-point number.

Return value

x	litodp (x)

lltodp (GCC GHS only)

Function name lltodp

Function type double lltodp (long long int x)

Description Returns the result of converting a 64-bit signed integer into a double-precision floating-point number.

Return value

x	lltodp (x)

dptoli

Function name dptoli

Function type int dptoli (double x)

Description Returns the result of converting a double-precision floating-point number into a signed integer.

Return value

x	dptoli(x)
$x >= 2^{31}$	0x7fffffff
$x <= -(2^{31})$	0x80000000
x = NaN	0x80000000

ultodp

Function name ultodp

Function type double ultodp (unsigned int x)

Description Returns the result of converting an unsigned integer into a double-precision floating-point number.

Return value

x	ultodp (x)

ulltodp (GCC GHS only)

Function name ulltodp

Function type double ulltodp (unsigned long long int x)

Description Returns the result of converting a 64-bit unsigned integer into a double-precision floating-point number.

Return value

x	ulltodp (x)

dptoul

Function name dptoul

Function type unsigned int dptoul (double x)

Description Returns the result of converting a double-precision floating-point number into an unsigned integer.

Return value

x	dptoul(x)
$x >= 2^{32}$	0xffffffff
$x <= -(2^{32})$	0x80000000
x = NaN	0x80000000

dptoull (GCC GHS only)

Function name dptoull

Function type unsigned long long int dptoull (double x)

Description Returns the result of converting a double-precision floating-point number into a 64-bit unsigned integer.

Return value

x	dptoull(x)
$x >= (2^{63})$	0x7fffffffffffffff
$x <= -(2^{64})$	0x8000000000000000
x = NaN	0x8000000000000000

litofp

Function name litofp

Function type float litofp (int x)

Description Returns the result of converting a signed integer into a single-precision floating-point number.

Return value

x	litofp (x)

fptoli

Function name fptoli

Function type int fptoli (float x)

Description Returns the result of converting a single-precision floating-point number into a signed integer.

Return value

x	fptoli(x)
$x >= 2^{31}$	0x7fffffff
$x <= -(2^{31})$	0x80000000
x = NaN	0x80000000

ultofp

Function name ultofp

Function type float ultofp (unsigned int x)

Description Returns the result of converting an unsigned integer into a single-precision floating-point number.

Return value

x	ultofp (x)

fptoul

Function name fptoul

Function type unsigned int fptoul (float x)

Description Returns the result of converting a single-precision real number into an unsigned integer.

Return value

x	fptoul(x)
$x >= 2^{32}$	0xffffffff
$x <= -(2^{32})$	0x80000000
x = NaN	0x80000000

dptoll (GCC GHS only)

Function name dptoll

Function type long long int dptoll (double x)

Description Returns the result of converting a double-precision floating-point number into a 64-bit signed integer.

Return value

x	dptoll(x)
$x >= 2^{63}$	0x7fffffffffffffff
$x <= -(2^{63})$	0x8000000000000000
x = NaN	0x8000000000000000

fptoll (GCC GHS only)

Function name fptoll

Function type long long int fptoll (float x)

Description Returns the result of converting a single-precision floating-point number into a 64-bit signed integer.

Return value

x	fptoll(x)
$x >= 2^{63}$	0x7fffffffffffffff
$x <= -(2^{63})$	0x8000000000000000
x = NaN	0x8000000000000000

fptoull (GCC GHS only)

Function name fptoull

Function type unsigned long long int fptoull (float x)

Description Returns the result of converting a single-precision floating-point number into a 64-bit unsigned integer.

Return value

x	fptoull(x)
$x >= (2^{63})$	0x7fffffffffffffff
$x <= -(2^{64})$	0x8000000000000000
x = NaN	0x8000000000000000

lltofp (GCC GHS only)

Function name lltofp

Function type float lltofp (long long int x)

Description Returns the result of converting a 64-bit signed integer into a single-precision floating-point number.

Return value

x	lltofp (x)

ulltofp (GCC GHS only)

Function name ulltofp

Function type float ulltofp (unsigned long long int x)

Description Returns the result of converting a 64-bit unsigned integer into a single-precision floating-point number.

Return value

x	ulltofp (x)

Comparison

dpcmp

Function name dpcmp

Function type int dpcmp (double x , double y)

Description Returns the result of comparing x with y.

Return value

x,y	dpcmp(x,y)
x < y x = y x > y GCC: x or y = NaN GHS: NaN	-1 0 1 1 Compares the bit pattern as an integer type.

In the GHS compiler, a == b is called by dcmp(b, a).

fpcmp

Function name fpcmp

Function type int fpcmp (float x , float y)

Description Returns the result of comparing x with y.

Return value

x,y	fpcmp(x,y)
x < y x = y x > y GCC: x or y = NaN GHS: NaN	-1 0 1 1 Compares the bit pattern as an integer type.

In the GHS compiler, a == b is called by fcmp(b, a).

4.2.2. Fundamental functions

Double precision

fabs

Function name fabs

Function type double fabs (double x)

Description Returns the result of changing the sign of x to positive.

Return value

x	fabs(x)
x = -NaN	+NaN
x = +NaN	+NaN

ceil

Function name ceil

Function type double ceil (double x)

Description Returns the minimum integer greater than x.

Return value

x	ceil(x)
x = NaN	NaN
x = +INF	+INF
x = -INF	-INF

floor

Function name floor

Function type double floor (double x)

Description Returns the maximum integer smaller than x.

Return value

x	floor(x)
x = NaN	NaN
x = +INF	+INF
x = -INF	-INF

fmod

Function name fmod

Function type double fmod (double x , double y)

Description Returns the result of dividing x by y.

Return value

	y				
x	NaN	+INF	-INF	+0	-0
NaN	NaN	NaN	NaN	NaN	NaN
+INF	NaN	NaN	NaN	NaN	NaN
-INF	NaN	NaN	NaN	NaN	NaN
+0	NaN	0	0	NaN	NaN
-0	NaN	0	0	NaN	NaN

modf

Function name modf

Function type double modf (double x , double *y)

Description Divides x into the integer block and floating-point block. Returns the floating-point block and assigns the integer block to *y.

Return value

x	modf(x, *y)
x = NaN	NaN , *y=NaN
x = +INF	0 , *y=+INF
x = -INF	0 , *y=-INF

frexp

Function name frexp

Function type double frexp (double x , int *y)

Description Returns the value of a when x is represented in the format $a*(2^b)$ ($0.5 <= |a| < 1$). Assigns the value of b to *y.

Return value

x	frexp(x, *y)
x = NaN	NaN, *y = 0
x = INF	INF, *y = 0
x = 0	0, *y = 0

ldexp

Function name ldexp

Function type double ldexp (double x , int y)

Description Returns the result of $x * 2^y$.

Return value

x	ldexp(x, y)
x = NaN	NaN
x = +INF	+INF
x = -INF	-INF

sqrt

Function name sqrt

Function type double sqrt (double x)

Description Returns the square root of x.

Return value

x	sqrt(x)
x < 0	NaN
x = NaN	NaN
x = +INF	+INF
x = -0	0

Single precision

fabsf

Function name fabsf

Function type float fabsf (float x)

Description Changes the sign of x to positive.

Return value

x	fabsf(x)
x = -NaN	+NaN
x = +NaN	+NaN

ceilf

Function name ceilf

Function type float ceilf (float x)

Description Returns the minimum integer value greater than x.

Return value

x	ceilf(x)
x = NaN	NaN
x = +INF	+INF
x = -INF	-INF

floorf

Function name floorf

Function type float floorf (float x)

Description Returns the maximum integer value smaller than x.

Return value

x	floorf(x)
x = NaN	NaN
x = +INF	+INF
x = -INF	-INF

fmodf

Function name fmodf

Function type float fmodf (float x , float y)

Description Returns the result of dividing x by y.

Return value

	y				
x	NaN	+INF	-INF	+0	-0
NaN	NaN	NaN	NaN	NaN	NaN
+INF	NaN	NaN	NaN	NaN	NaN
-INF	NaN	NaN	NaN	NaN	NaN
+0	NaN	0	0	NaN	NaN
-0	NaN	0	0	NaN	NaN

modff

Function name modff

Function type float modff (float x , float *y)

Description Divides x into the integer block and floating-point block. Returns the floating-point block and assigns the integer block to *y.

Return value

x	modff(x, *y)
x = NaN	NaN , *y=NaN
x = +INF	0 , *y=+INF
x = -INF	0 , *y=-INF

frexpf

Function name frexpf

Function type float frexpf (float x , int *y)

Description Returns the value of a when x is represented in the format $a*(2^b)$ (0.5 <= |a| < 1). Assigns the value of b to *y.

Return value

x	frexpf(x, *y)
x = NaN	NaN, *y = 0
x = INF	INF *y = 0
x = 0	0 *y = 0

ldexpf

Function name ldexpf

Function type float ldexpf (float x , int y)

Description Returns the result of x * (2^y).

Return value

x ,y	ldexpf(x, y)
x = NaN	NaN
x = +INF	+INF
x = -INF	-INF

sqrtf

Function name sqrtf

Function type float sqrtf (float x)

Description Returns the square root of x.

Return value

x	sqrtf(x)
x < 0	NaN
x = NaN	NaN
x = +INF	+INF
x = -0	0

4.2.3. Transcendental functions

Double precision

asin

Function name asin

Function type double asin (double x)

Description Returns the inverse sine of x.

Return value

x	asin(x)
$\|x\| < 2^{-1022}$	x
|x| > 1	NaN
x = NaN	NaN

acos

Function name acos

Function type double acos (double x)

Description Returns the inverse cosine of x.

Return value

x	acos(x)
|x| > 1	NaN
x = NaN	NaN

atan

Function name atan

Function type double atan (double x)

Description Returns the inverse tangent of x.

Return value

x	atan(x)
$\|x\| < 2^{-1022}$	x
x = NaN	NaN

atan2

Function name atan2

Function type double atan2 (double x , double y)

Description Returns the inverse tangent of x/y.

Return value

x,y	atan2(x,y)
x = NaN	NaN
y = NaN	NaN
x = INF and y = INF	NaN
x = 0 and y=0	NaN

cos

Function name cos

Function type double cos (double x)

Description Returns the cosine of x.

Return value

x	cos(x)
$\|x\| >= 2^{64}$	NaN
x = NaN	NaN

cosh

Function name cosh

Function type double cosh (double x)

Description Returns the hyperbolic cosine of x.

Return value

x	cosh(x)
x = NaN	NaN
x = +INF	+INF
x = -INF	+INF

exp

Function name exp

Function type double exp (double x)

Description Returns the result of e^x.

Return value

x	exp(x)
$\|x\| < 2^{-1023}$	1
x = NaN	NaN
x = +INF	+INF
x = -INF	0

log

Function name log

Function type double log (double x)

Description Returns the natural logarithm of x.

Return value

x	log(x)
x < 0	NaN
x = NaN	NaN
x = +INF	+INF
x = 0	-INF

log10

Function name log10

Function type double log10 (double x)

Description Returns the common logarithm of x.

Return value

x	log10(x)
x < 0	NaN
x = NaN	NaN
x = +INF	+INF
x = 0	-INF

pow

Function name pow

Function type double pow (double x , double y)

Description Returns the result of x^y.

Return value

	y						
x	NaN	+INF	-INF	+0	-0	+1	-1
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
+1	NaN	+1	+1	+1	+1	x	+1
+0	NaN	0	+INF	+1	+1	x	+INF
-0	NaN	0	+INF	+1	+1	x	+INF
-1	NaN	NaN	NaN	+1	+1	x	-1
+INF	NaN	+INF	0	+1	+1	x	0
-INF	NaN	NaN	0	+1	+1	x	0
1 < x	NaN	+INF	0	+1	+1	x	pow(x)
0 < x < 1	NaN	0	+INF	+1	+1	x	pow(x)
-1 < x < 0	NaN	0	NaN	+1	+1	x	pow(x)
x < -1	NaN	NaN	0	+1	+1	x	pow(x)

	y					
x	y <-1(int)	y <-1(!int)	-1 < y < 0	0 < y < 1	1 < y(int)	1 < y(!int)
NaN	NaN	NaN	NaN	NaN	NaN	NaN
+1	+1	+1	+1	+1	+1	+1
+0	+INF	+INF	+INF	0	0	0
-0	+INF	+INF	+INF	0	0	0
-1	pow(x)	NaN	NaN	NaN	pow(x)	NaN
+INF	0	0	0	+INF	+INF	+INF
-INF	0	0	0	NaN	pow(x)	NaN
1 < x	pow(x)	pow(x)	pow(x)	pow(x)	pow(x)	pow(x)
0 < x < 1	pow(x)	pow(x)	pow(x)	pow(x)	pow(x)	pow(x)
-1 < x < 0	pow(x)	NaN	NaN	NaN	pow(x)	NaN
x < -1	pow(x)	NaN	NaN	NaN	pow(x)	NaN

sin

Function name sin

Function type double sin (double x)

Description Returns the sine of x.

Return value

x	sin(x)
$\|x\| >= 2^{64}$	NaN
x = NaN	NaN

sinh

Function name sinh

Function type double sinh (double x)

Description Returns the hyperbolic sine of x.

Return value

x	sinh(x)
x = NaN	NaN
x = +INF	+INF
x = -INF	-INF

tan

Function name tan

Function type double tan (double x)

Description Returns the tangent of x.

Return value

x	tan(x)
$\|x\| >= 2^{64}$	NaN
x = NaN	NaN

tanh

Function name tanh

Function type double tanh (double x)

Description Returns the hyperbolic tangent of x.

Return value

x	tanh(x)
$\|x\| < 2^{-1022}$	0
$\|x\| > 2^{9}$	Signed 1 of x
x = NaN	NaN
x = +INF	1
x = -INF	-1

Single precision

acosf

Function name acosf

Function type float acosf (float x)

Description Returns the inverse cosine of x.

Return value

x	acosf(x)
|x| > 1	NaN
x = NaN	NaN

asinf

Function name asinf

Function type float asinf (float x)

Description Returns the inverse sine of x.

Return value

x	asinf(x)
|x| > 1	NaN
x = NaN	NaN

atanf

Function name atanf

Function type float atanf (float x)

Description Returns the inverse tangent of x.

Return value

x	atanf(x)
x = NaN	NaN

atan2f

Function name atan2f

Function type float atan2f (float x , float y)

Description Returns the inverse tangent of x/y.

Return value

x,y	atan2f(x,y)
x = NaN	NaN
y = NaN	NaN
x = INF and y = INF	NaN
x = 0 and y = 0	NaN

cosf

Function name cosf

Function type float cosf (float x)

Description Returns the cosine of x.

Return value

x	cosf(x)
$\|x\| >= 2^{31}$	NaN
x = NaN	NaN

coshf

Function name coshf

Function type float coshf (float x)

Description Returns the hyperbolic cosine of x.

Return value

x	coshf(x)
x = NaN	NaN
x = +INF	+INF
x = -INF	+INF

expf

Function name expf

Function type float expf (float x)

Description Returns the result of e^x.

Return value

x	expf(x)
$\|x\| < 2^{-127}$	1
x = NaN	NaN
x = +INF	+INF
x = -INF	0

logf

Function name logf

Function type float logf (float x)

Description Returns the natural logarithm of x.

Return value

x	logf(x)
x < 0	NaN
x = NaN	NaN
x = +INF	+INF
x = 0	-INF

log10f

Function name log10f

Function type float log10f (float x)

Description Returns the common logarithm of x.

Return value

x	logf10(x)
x < 0	NaN
x = NaN	NaN
x = +INF	+INF
x = 0	-INF

powf

Function name powf

Function type float powf (float x , float y)

Description Returns the result of x^y.

Return value

x	y NaN	+INF	-INF	+0	-0	+1	-1
NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
+1	NaN	+1	+1	+1	+1	x	+1
+0	NaN	0	+INF	+1	+1	x	+INF
-0	NaN	0	+INF	+1	+1	x	+INF
-1	NaN	NaN	NaN	+1	+1	x	-1
+INF	NaN	+INF	0	+1	+1	x	0
-INF	NaN	NaN	0	+1	+1	x	0
1 < x	NaN	+INF	0	+1	+1	x	powf(x)
0 < x < 1	NaN	0	+INF	+1	+1	x	powf(x)
-1 < x < 0	NaN	0	NaN	+1	+1	x	powf(x)
x < -1	NaN	NaN	0	+1	+1	x	powf(x)

x	y y <-1(int)	y <-1(!int)	-1 < y < 0	0 < y < 1	1 < y(int)	1 < y(!int)
NaN	NaN	NaN	NaN	NaN	NaN	NaN
+1	+1	+1	+1	+1	+1	+1
+0	+INF	+INF	+INF	0	0	0
-0	+INF	+INF	+INF	0	0	0
-1	powf(x)	NaN	NaN	NaN	powf(x)	NaN
+INF	0	0	0	+INF	+INF	+INF
-INF	0	0	0	NaN	powf(x)	NaN
1 < x	powf(x)	powf(x)	powf(x)	powf(x)	powf(x)	powf(x)
0 < x < 1	powf(x)	powf(x)	powf(x)	powf(x)	powf(x)	powf(x)
-1 < x < 0	powf(x)	NaN	NaN	NaN	powf(x)	NaN
x < -1	powf(x)	NaN	NaN	NaN	powf(x)	NaN

sinf

Function name sinf

Function type float sinf (float x)

Description Returns the sine of x.

Return value

x	sinf(x)
$\|x\| >= 2^{31}$	NaN
x = NaN	NaN

sinhf

Function name sinhf

Function type float sinhf (float x)

Description Returns the hyperbolic sine of x.

Return value

x	sinhf(x)
x = NaN	NaN
x = +INF	+INF
x = -INF	-INF

tanf

Function name tanf

Function type float tanf (float x)

Description Returns the tangent of x.

Return value

x	tanf(x)
$\|x\| >= 2^{31}$	NaN
x = NaN	NaN

tanhf

Function name tanhf

Function type float tanhf (float x)

Description Returns the hyperbolic tangent of x.

Return value

x	tanhf(x)
$\|x\| < 2^{-126}$	0
$\|x\| > 2^{9}$	Signed 1 of x
x = NaN	NaN
x = +INF	1
x = -INF	-1

4.2.4. Additional functions

isnan

Function name isnan

Function type int isnan (double x)

Description Returns true if x = NaN.

Return value

x	isnan (x)
x = +NaN	1
x = -NaN	1
x != NaN	0

isnanf

Function name isnanf

Function type int isnanf (float x)

Description Returns true if x = NaN.

Return value

x	isnanf (x)
x = +NaN	1
x = -NaN	1
x != NaN	0

isinf

Function name isinf

Function type int isinf (double x)

Description Returns true if x = INF.

Return value

x	isinf (x)
x != INF	0
GCC: x = INF	1
GHS: x = -INF	-1
GHS: x = +INF	+1

isinff

Function name isinff

Function type int isinff (float x)

Description Returns true if x = INF.

Return value

x	isinff (x)
x = INF	1
GCC: x = INF	1
GHS: x = -INF	-1
GHS: x = +INF	+1

4.2.5. GCC runtime function

__negdf2 (GCC only)

Function name __negdf2 (GCC only)

Function type float __negdf2 (float x)

Description Returns the result of inverting the sign of x.

Return value

x	__negdf2 (x)
x = +0	-0
x = -0	+0
x = +NaN	-NaN
x = -NaN	+NaN
x = +INF	-INF
x = -INF	+INF

__negsf2 (GCC only)

Function name __negsf2 (GCC only)

Function type float __negsf2 (float x)

Description Returns the result of inverting the sign of x.

Return value

x	__negsf2 (x)
x = +0	-0
x = -0	+0
x = +NaN	-NaN
x = -NaN	+NaN
x = +INF	-INF
x = -INF	+INF

__eqdf2, __nedf2, __ltdf2, __ledf2 (GCC only)

Function name __eqdf2, __nedf2, __ltdf2, __ledf2 (GCC only)

Function type int __eqdf2(double x, double y)

int __nedf2(double x, double y)

int __ltdf2(double x, double y)

int __ledf2(double x, double y)

Description Returns the result of comparing x with y.

Return value

x,y	<FUNC> (x,y)
x < y	-1
x = y	0
x > y	1
x or y = NaN	1

_gtdf2, __gedf2 (GCC only)

Function name __gtdf2, __gedf2 (GCC only)

Function type int __gtdf2(double x, double y)

int __gedf2(double x, double y)

Description Returns the result of comparing x with y.

Return value

x, y	<FUNC> (x,y)
x < y	-1
x = y	0
x > y	1
x or y = NaN	-1

__eqsf2, __nesf2, __ltsf2, __lesf2 (GCC only)

Function name __eqsf2, __nesf2, __ltsf2, __lesf2 (GCC only)

Function type int __eqsf2(float x, float y)

int __nesf2(float x, float y)

int __ltsf2(float x, float y)

int __lesf2(float x, float y)

Description Returns the result of comparing x with y.

Return value

x,y	<FUNC> (x,y)
x < y	-1
x = y	0
x > y	1
x or y = NaN	1

__gtsf2, __gesf2 (GCC only)

Function name __gtsf2, __gesf2 (GCC only)

Function type int __gtsf2(float x, float y)

int __gesf2(float x, float y)

Description Returns the result of comparing x with y.

Return value

x,y	<FUNC> (x,y)
x < y	-1
x = y	0
x > y	1
x or y = NaN	-1

4.3. Macros

The following macros are defined.

float.h

For small numbers

FLT_RADIX, FLT_ROUNDS

For **float** type

FLT_MANT_DIG, FLT_EPSILON, FLT_DIG, FLT_MIN_EXP, FLT_MIN, FLT_MIN_10_EXP, FLT_MAX_EXP, FLT_MAX, FLT_MAX_10_EXP

For **double** type

DBL_MANT_DIG, DBL_EPSILON, DBL_DIG, DBL_MIN_EXP, DBL_MIN, DBL_MIN_10_EXP, DBL_MAX_EXP, DBL_MAX, DBL_MAX_10_EXP

For **long double** type

LDBL_MANT_DIG, LDBL_EPSILON, LDBL_DIG, LDBL_MIN_EXP, LDBL_MIN, LDBL_MIN_10_EXP, LDBL_MAX_EXP, LDBL_MAX, LDBL_MAX_10_EXP

math.h

HUGE_VAL Sets the positive infinit.

Constant macro The following macros are defined.

Macro	Description
M_E	*e*
M_LOG2E	*log 2e*
M_LOG10E	*log 10e*
M_LN2	*log e2*
M_LN10	*log e10*
M_PI	*π*
M_PI_2	*π / 2*
M_PI_4	*π / 4*
M_1_PI	*1 / π*
M_2_PI	*2 / π*
M_2_SQRTPI	*2 / sqrt (π)*
M_SQRT2	*sqrt(2)*
M_SQRT1_2	*1 / sqrt(2)*

INDEX