

The Cygnus C Math Library

libm 1.4
May 1993

Steve Chamberlain
Roland Pesch
Cygnus Support

Libm

Cygnus Support
sac@cygnus.com
doc@cygnus.com

Copyright © 1992, 1993, 1994, 1995, 1996 Cygnus Support

'libm' includes software developed by the University of California, Berkeley and its contributors.

'libm' includes software developed by Martin Jackson, Graham Haley and Steve Chamberlain of Tadpole Technology and released to Cygnus.

'libm' includes software developed at SunPro, a Sun Microsystems, Inc. business. Permission to use, copy, modify, and distribute this software is freely granted, provided that this notice is preserved.

Permission is granted to make and distribute verbatim copies of this manual provided the copyright notice and this permission notice are preserved on all copies.

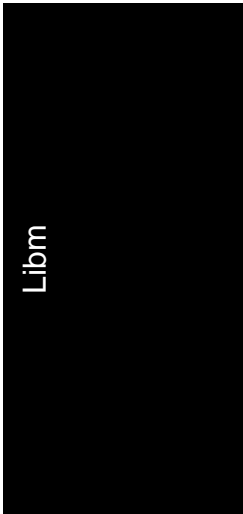
Permission is granted to copy and distribute modified versions of this manual under the conditions for verbatim copying, subject to the terms of the GNU General Public License, which includes the provision that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this manual into another language, under the above conditions for modified versions.

Table of Contents

1 Reentrancy Properties of libm..... 1

Index 3



1 Reentrancy Properties of libm

When a math function detects an error, it sets the static variable `errno`. Depending upon the severity of the error, it may also print a message to `stderr`. None of this behavior is reentrant. When one process detects an error, it sets `errno`. If another process is testing `errno`, it detects the change and probably fails. Note that failing system calls can also set `errno`. This problem can only be fixed by either ignoring `errno`, or treating it as part of the context of a process and switching it along with the rest of a processor state. In normal debugged programs, there are usually no math subroutine errors—and therefore no `matherr` calls; in that situation, the math functions behave reentrantly.

As an alternative, you can use the reentrant versions of the mathematical functions: these versions have a different name, normally formed by prepending ‘_’ and appending ‘_r’, and use an extra argument—a pointer to the particular reentrancy structure to use. See section “Reentrancy” in *The Cygnus C Support Library*, for more discussion of this approach to reentrancy.

The reentrancy structure is always an additional first argument; for example, the reentrant version of ‘double `acos` (double `x`)’ is ‘double `_acos_r` (void *`reent`, double `x`)’.

Here is a list of the names for reentrant versions of the mathematical library functions:

<code>_acos_r</code>	<code>_gammaf_r_r</code>	<code>_log10f_r</code>
<code>_acosf_r</code>	<code>_hypot_r</code>	<code>_log_r</code>
<code>_acosh_r</code>	<code>_hypotf_r</code>	<code>_logf_r</code>
<code>_acoshf_r</code>	<code>_j0_r</code>	<code>_pow_r</code>
<code>_asin_r</code>	<code>_j0f_r</code>	<code>_powf_r</code>
<code>_asinf_r</code>	<code>_j1_r</code>	<code>_remainder_r</code>
<code>_atanh_r</code>	<code>_j1f_r</code>	<code>_sinh_r</code>
<code>_atanhf_r</code>	<code>_jn_r</code>	<code>_sinhf_r</code>
<code>_cosh_r</code>	<code>_jnf_r</code>	<code>_sqrt_r</code>
<code>_coshf_r</code>	<code>_ldexp_r</code>	<code>_sqrtf_r</code>
<code>_exp_r</code>	<code>_ldexpf_r</code>	<code>_y0_r</code>
<code>_expf_r</code>	<code>_lgamma_r</code>	<code>_y0f_r</code>
<code>_fmod_r</code>	<code>_lgamma_r_r</code>	<code>_y1_r</code>
<code>_fmodf_r</code>	<code>_lgammaf_r</code>	<code>_y1f_r</code>
<code>_gamma_r</code>	<code>_lgammaf_r_r</code>	<code>_yn_r</code>
<code>_gamma_r_r</code>	<code>_log10_r</code>	<code>_ynf_r</code>
<code>_gammaf_r</code>		

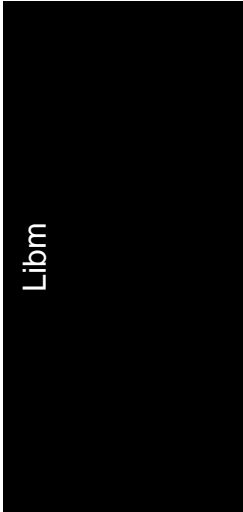
Index

M

matherr and reentrancy..... 1

R

reentrancy..... 1



The body of this manual is set in
pncr at 10.95pt,
with headings in **pncb at 10.95pt**
and examples in pcr.rr.
pncrri at 10.95pt and
pcrro
are used for emphasis.