INTERNATIONAL STANDARD

ISO/IEC 24747

First edition
2009-01-15

Information technology — Programming languages, their environments and system software interfaces — Extensions to the C Library to support mathematical special functions

Technologies de l'information — Langages de programmation, leur environnement et interfaces des logiciels de systèmes — Extensions à la bibliothèque C pour supporter les fonctions mathématiques spéciales

Reference number
ISO/IEC 24747:2009(E)

© ISO/IEC 2009

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2009

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

--`,,```,,,,````-`-`,,`,,`,`,,`---

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

© ISO/IEC 2009 – All rights reserved

Not for Resale

Contents

Contents iii

List of Tables v

Foreword vi

Introduction vii

1 Scope 1
1.1 Relation to C Standard Library Introduction . . . 1
1.2 Categories of extensions . . . 1

2 Normative references 3

3 Terms, definitions, and symbols 5

4 Conformance 7

5 Predefined macro names 9

6 Mathematical special functions 11
6.1 Standard headers . . . 11
6.2 Additions to header `<math.h>` . . . 11
6.2.1 associated Laguerre polynomials . . . 14
6.2.2 associated Legendre polynomials . . . 14
6.2.3 beta function . . . 15
6.2.4 (complete) elliptic integral of the first kind . . . 15
6.2.5 (complete) elliptic integral of the second kind . . . 15
6.2.6 (complete) elliptic integral of the third kind . . . 16
6.2.7 regular modified cylindrical Bessel functions . . . 16
6.2.8 cylindrical Bessel functions (of the first kind) . . . 16
6.2.9 irregular modified cylindrical Bessel functions . . . 17
6.2.10 cylindrical Neumann functions . . . 17
6.2.11 (incomplete) elliptic integral of the first kind . . . 17
6.2.12 (incomplete) elliptic integral of the second kind . . . 18

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

6.2.13 (incomplete) elliptic integral of the third kind 18
6.2.14 exponential integral 18
6.2.15 Hermite polynomials 19
6.2.16 Laguerre polynomials 19
6.2.17 Legendre polynomials 19
6.2.18 Riemann zeta function 20
6.2.19 spherical Bessel functions (of the first kind) 20
6.2.20 spherical associated Legendre functions 20
6.2.21 spherical Neumann functions 21
6.3 Additions to header `<tgmath.h>` 21

Bibliography **23**

Index **25**

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

List of Tables

1 Numerical library summary 1

2 Additions to header <math.h> synopsis 14
3 Additions to header <tgmath.h> synopsis 22

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 24747 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interface*s.

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Introduction

This International Standard is divided into three major subdivisions:

— preliminary elements (clauses 1 - 4);

— indicating conformance (clause 5);

— the library facilities (clause 6).

Footnotes are provided to emphasize consequences of the rules described in that subclause or elsewhere in this International Standard. References are used to refer to other related documents and subclauses. A bibliography lists documents that were referred to during the preparation of this International Standard.

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Information technology — Programming languages, their environments and system software interfaces — Extensions to the C Library to support mathematical special functions

1 Scope [scop]

1.1 Relation to C Standard Library Introduction [descr]

1 This International Standard defines extensions to the *C standard library* that is defined in the International Standard for the C programming language, see Clause 2.

2 Unless otherwise specified, the whole of the *C* Standard Library is included into this International Standard by reference, see Clause 2.

1.2 Categories of extensions [exten]

1 This International Standard defines library extensions to the C Standard Library to support Mathematical Special functions to be added to `<math.h>` and `<tgmath.h>`.

Table 1: Numerical library summary

Subclause	Header(s)
6.2 Additions to	`<math.h>`
6.3 Additions to	`<tgmath.h>`

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

(Blank page)

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

2 Normative references [norm]

1 The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

2 ISO/IEC 9899:1999, *Programming languages — C*

3 ISO/IEC 9899:1999/Cor. 1:2001, *Programming languages — C — Technical Corrigendum 1*

4 ISO/IEC 9899:1999/Cor. 2:2004, *Programming languages — C — Technical Corrigendum 2*

5 ISO/IEC 9899:1999/Cor. 3:2007, *Programming languages — C — Technical Corrigendum 3*

6 ISO/IEC 2382-1:1993, *Information technology — Vocabulary — Part 1: Fundamental terms*

--`,,```,,,,````-`-`,,`,,`,`,,`---
© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

(Blank page)

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

3 Terms, definitions, and symbols [terms]

1 For the purposes of this document, the terms and definitions given in ISO/IEC 9899:1999 and the associated Technical Corrigenda and ISO/IEC 2382-1 apply, see Clause 2.

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

(Blank page)

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

4 Conformance [confor]

1 If a "shall" requirement is violated, the behavior is undefined.

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

(Blank page)

-`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

5 Predefined macro names [pred]

1 The following macro name is conditionally defined by the implementation:
`__STDC_MATH_SPEC_FUNCS__` The integer constant `200808`, intended to indicate conformance to this International Standard. [1)]

[1)] The intention is that this will remain an integer constant of type `long int` that is increased with each revision of this International Standard.

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

(Blank page)

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

6 Mathematical special functions [num.sf]

6.1 Standard headers [num.sf.header]

1 The functions declared in Clause 6 and its subclauses are not declared by their respective header if `__STDC_WANT_MATH_SPEC_FUNCS__` is defined as a macro which expands to the integer constant `0` at the point in the source file where the appropriate header is included.

2 The functions declared in Clause 6 and its subclauses are declared by their respective headers if `__STDC_WANT_MATH_SPEC_FUNCS__` is defined as a macro which expands to the integer constant 1 at the point in the source file where the appropriate header is included. [2)]

3 Functions declared in Clause 6 and its subclauses shall not be declared by their respective headers if `__STDC_WANT_MATH_SPEC_FUNCS__` is not defined as a macro at the point in the source file where the appropriate header is included.

4 Within a preprocessing translation unit, `__STDC_WANT_MATH_SPEC_FUNCS__` shall be defined identically for all inclusions of any headers from Clause 6. If `__STDC_WANT_MATH_SPEC_FUNCS__` is defined differently for any such inclusion, the implementation shall issue a diagnostic as if a preprocessor error directive was used.

6.2 Additions to header `<math.h>` [num.sf.math]

1 Table 2 summarizes the functions that are added to header `<math.h>`. The detailed signatures are given in the synopsis.

2 Each of these functions is provided for arguments of type `float`, `double`, and `long double`. The signatures added to header `<math.h>` are:

```
// [6.2.1] associated Laguerre polynomials:
double       assoc_laguerre(unsigned n, unsigned m, double x);
float        assoc_laguerref(unsigned n, unsigned m, float x);
long double  assoc_laguerrel(unsigned n, unsigned m, long double x);

// [6.2.2] associated Legendre polynomials:
double       assoc_legendre(unsigned l, unsigned m, double x);
float        assoc_legendref(unsigned l, unsigned m, float x);
long double  assoc_legendrel(unsigned l, unsigned m, long double x);

// [6.2.3] beta function:
double       beta(double x, double y);
float        betaf(float x, float y);
long double  betal(long double x, long double y);
```

[2)] Future revisions of this International Standard may define meanings for other values of `__STDC_WANT_MATH_SPEC_FUNCS__`.

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

```
// [6.2.4] (complete) elliptic integral of the first kind:
double       comp_ellint_1(double k);
float        comp_ellint_1f(float k);
long double  comp_ellint_1l(long double k);

// [6.2.5] (complete) elliptic integral of the second kind:
double       comp_ellint_2(double k);
float        comp_ellint_2f(float k);
long double  comp_ellint_2l(long double k);

// [6.2.6] (complete) elliptic integral of the third kind:
double       comp_ellint_3(double k, double nu);
float        comp_ellint_3f(float k, float nu);
long double  comp_ellint_3l(long double k, long double nu);

// [6.2.7] regular modified cylindrical Bessel functions:
double       cyl_bessel_i(double nu, double x);
float        cyl_bessel_if(float nu, float  x);
long double  cyl_bessel_il(long double nu, long double x);

// [6.2.8] cylindrical Bessel functions (of the first kind):
double       cyl_bessel_j(double nu, double x);
float        cyl_bessel_jf(float nu, float  x);
long double  cyl_bessel_jl(long double nu, long double x);

// [6.2.9] irregular modified cylindrical Bessel functions:
double       cyl_bessel_k(double nu, double x);
float        cyl_bessel_kf(float nu, float  x);
long double  cyl_bessel_kl(long double nu, long double x);

// [6.2.10] cylindrical Neumann functions;
// cylindrical Bessel functions (of the second kind):
double       cyl_neumann(double nu, double x);
float        cyl_neumannf(float nu, float  x);
long double  cyl_neumannl(long double nu, long double x);

// [6.2.11] (incomplete) elliptic integral of the first kind:
double       ellint_1(double k, double phi);
float        ellint_1f(float k, float phi);
long double  ellint_1l(long double k, long double phi);

// [6.2.12] (incomplete) elliptic integral of the second kind:
double       ellint_2(double k, double phi);
float        ellint_2f(float k, float phi);
long double  ellint_2l(long double k, long double phi);

// [6.2.13] (incomplete) elliptic integral of the third kind:
double       ellint_3(double k, double nu, double phi);
float        ellint_3f(float k, float nu, float phi);
long double  ellint_3l(long double k, long double nu, long double phi);
```


Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale
© ISO/IEC 2009 – All rights reserved

```
// [6.2.14] exponential integral:
double       expint(double x);
float        expintf(float x);
long double  expintl(long double x);

// [6.2.15] Hermite polynomials:
double       hermite(unsigned n, double x);
float        hermitef(unsigned n, float x);
long double  hermitel(unsigned n, long double x);

// [6.2.16] Laguerre polynomials:
double       laguerre(unsigned n, double x);
float        laguerref(unsigned n, float x);
long double  laguerrel(unsigned n, long double x);

// [6.2.17] Legendre polynomials:
double       legendre(unsigned l, double x);
float        legendref(unsigned l, float x);
long double  legendrel(unsigned l, long double x);

// [6.2.18] Riemann zeta function:
double       riemann_zeta(double);
float        riemann_zetaf(float);
long double  riemann_zetal(long double);

// [6.2.19] spherical Bessel functions (of the first kind):
double       sph_bessel(unsigned n, double x);
float        sph_besself(unsigned n, float  x);
long double  sph_bessell(unsigned n, long double x);

// [6.2.20] spherical associated Legendre functions:
double       sph_legendre(unsigned l, unsigned m, double theta);
float        sph_legendref(unsigned l, unsigned m, float theta);
long double  sph_legendrel(unsigned l, unsigned m, long double theta);

// [6.2.21] spherical Neumann functions;
// spherical Bessel functions (of the second kind):
double       sph_neumann(unsigned n, double x);
float        sph_neumannf(unsigned n, float  x);
long double  sph_neumannl(unsigned n, long double x);
```

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

Table 2: Additions to header <math.h> synopsis

Functions:		
assoc_laguerre	cyl_bessel_j	hermite
assoc_legendre	cyl_bessel_k	legendre
beta	cyl_neumann	laguerre
comp_ellint_1	ellint_1	riemann_zeta
comp_ellint_2	ellint_2	sph_bessel
comp_ellint_3	ellint_3	sph_legendre
cyl_bessel_i	expint	sph_neumann

3 Each of the functions declared above shall return a NaN (Not a Number) if any argument value is a NaN, but it shall not report a domain error. Otherwise, each of the functions declared above shall report a domain error for just those argument values for which:

— the function description's Returns clause explicitly specifies a domain, and those arguments fall outside the specified domain; or

— the corresponding mathematical function value has a non-zero imaginary component; or

— the corresponding mathematical function is not mathematically defined.[3)]

4 Unless otherwise specified, a function is defined for all finite values, for negative infinity, and for positive infinity.

6.2.1 associated Laguerre polynomials [num.sf.Lnm]

```
double       assoc_laguerre(unsigned n, unsigned m, double x);
float        assoc_laguerref(unsigned n, unsigned m, float x);
long double  assoc_laguerrel(unsigned n, unsigned m, long double x);
```

1 *Effects:* These functions compute the associated Laguerre polynomials of their respective arguments `n`, `m`, and `x`.

2 *Returns:* The `assoc_laguerre` functions return

$$\mathsf{L}_n^m(x) = (-1)^m \frac{\mathrm{d}^m}{\mathrm{d}x^m} \mathsf{L}_{n+m}(x), \quad \text{for } x \geq 0.$$

3 *Note:* The effect of calling each of these functions is implementation-defined if `n >= 128` or if `m >= 128`.

6.2.2 associated Legendre polynomials [num.sf.Plm]

```
double       assoc_legendre(unsigned l, unsigned m, double x);
float        assoc_legendref(unsigned l, unsigned m, float x);
long double  assoc_legendrel(unsigned l, unsigned m, long double x);
```

[3)] A mathematical function is mathematically defined for a given set of argument values if it is explicitly defined for that set of argument values or if its limiting value exists and does not depend on the direction of approach.

--`,,```,,,,````-`-`,,`,,`,`,,`---

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS

© ISO/IEC 2009 – All rights reserved

Not for Resale

1 *Effects:* These functions compute the associated Legendre functions of their respective arguments `l`, `m`, and `x`.

2 *Returns:* The `assoc_legendre` functions return

$$\mathsf{P}_\ell^m(x) = (1-x^2)^{m/2}\,\frac{\mathrm{d}^m}{\mathrm{d}x^m}\,\mathsf{P}_\ell(x), \quad \text{for } |x| \le 1.$$

3 *Note:* The effect of calling each of these functions is implementation-defined if `l >= 128`.

6.2.3 beta function [num.sf.beta]

```
double       beta(double x, double y);
float        betaf(float x, float y);
long double  betal(long double x, long double y);
```

1 *Effects:* These functions compute the beta function of their respective arguments `x` and `y`.

2 *Returns:* The `beta` functions return

$$\mathsf{B}(x,y) = \frac{\Gamma(x)\,\Gamma(y)}{\Gamma(x+y)}, \quad \text{for } x > 0,\ y > 0.$$

6.2.4 (complete) elliptic integral of the first kind [num.sf.ellK]

```
double       comp_ellint_1(double k);
float        comp_ellint_1f(float k);
long double  comp_ellint_1l(long double k);
```

1 *Effects:* These functions compute the complete elliptic integral of the first kind of their respective arguments `k`.

2 *Returns:* The `comp_ellint_1` functions return

$$\mathsf{K}(k) = \mathsf{F}(k, \pi/2), \quad \text{for } |k| \le 1.$$

3 See 6.2.11.

6.2.5 (complete) elliptic integral of the second kind [num.sf.ellEx]

```
double       comp_ellint_2(double k);
float        comp_ellint_2f(float k);
long double  comp_ellint_2l(long double k);
```

1 *Effects:* These functions compute the complete elliptic integral of the second kind of their respective arguments `k`.

2 *Returns:* The `comp_ellint_2` functions return

$$\mathsf{E}(k) = \mathsf{E}(k, \pi/2), \quad \text{for } |k| \le 1.$$

3 See 6.2.12.

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

6.2.6 (complete) elliptic integral of the third kind [num.sf.ellPx]

```
double       comp_ellint_3(double k, double nu);
float        comp_ellint_3f(float k, float nu);
long double  comp_ellint_3l(long double k, long double nu);
```

1 *Effects:* These functions compute the complete elliptic integral of the third kind of their respective arguments `k` and `nu`.

2 *Returns:* The `comp_ellint_3` functions return

$$\Pi(\nu,k) = \Pi(\nu,k,\pi/2), \quad \text{for } |k| \le 1.$$

3 See 6.2.13.

6.2.7 regular modified cylindrical Bessel functions [num.sf.I]

```
double       cyl_bessel_i(double nu, double x);
float        cyl_bessel_if(float nu, float  x);
long double  cyl_bessel_il(long double nu, long double x);
```

1 *Effects:* These functions compute the regular modified cylindrical Bessel functions of their respective arguments `nu` and `x`.

2 *Returns:* The `cyl_bessel_i` functions return

$$\mathsf{I}_\nu(x) = \mathrm{i}^{-\nu}\mathsf{J}_\nu(\mathrm{i}x) = \sum_{k=0}^{\infty} \frac{(x/2)^{\nu+2k}}{k!\,\Gamma(\nu+k+1)}, \quad \text{for } x \ge 0.$$

3 *Note:* The effect of calling each of these functions is implementation-defined if `nu >= 128`.

6.2.8 cylindrical Bessel functions (of the first kind) [num.sf.J]

```
double       cyl_bessel_j(double nu, double x);
float        cyl_bessel_jf(float nu, float  x);
long double  cyl_bessel_jl(long double nu, long double x);
```

1 *Effects:* These functions compute the cylindrical Bessel functions of the first kind of their respective arguments `nu` and `x`.

2 *Returns:* The `cyl_bessel_j` functions return

$$\mathsf{J}_\nu(x) = \sum_{k=0}^{\infty} \frac{(-1)^k (x/2)^{\nu+2k}}{k!\,\Gamma(\nu+k+1)}, \quad \text{for } x \ge 0.$$

3 *Note:* The effect of calling each of these functions is implementation-defined if `nu >= 128`.

--`,,```,,,,````-`-`,,`,,`,`,,`---
© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

6.2.9 irregular modified cylindrical Bessel functions [num.sf.K]

```
double       cyl_bessel_k(double nu, double x);
float        cyl_bessel_kf(float nu, float  x);
long double  cyl_bessel_kl(long double nu, long double x);
```

1 *Effects:* These functions compute the irregular modified cylindrical Bessel functions of their respective arguments `nu` and `x`.

2 *Returns:* The `cyl_bessel_k` functions return

$$\mathsf{K}_\nu(x) = (\pi/2)\mathrm{i}^{\nu+1}(\mathsf{J}_\nu(\mathrm{i}x) + \mathrm{i}\mathsf{N}_\nu(\mathrm{i}x)) = \begin{cases} \dfrac{\pi}{2}\dfrac{\mathsf{I}_{-\nu}(x) - \mathsf{I}_\nu(x)}{\sin\nu\pi}, & \text{for } x \geq 0 \text{ and non-integral } \nu \\ \dfrac{\pi}{2}\lim\limits_{\mu\to\nu}\dfrac{\mathsf{I}_{-\mu}(x) - \mathsf{I}_\mu(x)}{\sin\mu\pi}, & \text{for } x \geq 0 \text{ and integral } \nu \end{cases}.$$

3 *Note:* The effect of calling each of these functions is implementation-defined if `nu >= 128`.

6.2.10 cylindrical Neumann functions [num.sf.N]

```
double       cyl_neumann(double nu, double x);
float        cyl_neumannf(float nu, float  x);
long double  cyl_neumannl(long double nu, long double x);
```

1 *Effects:* These functions compute the cylindrical Neumann functions, also known as the cylindrical Bessel functions of the second kind, of their respective arguments `nu` and `x`.

2 *Returns:* The `cyl_neumann` functions return

$$\mathsf{N}_\nu(x) = \begin{cases} \dfrac{\mathsf{J}_\nu(x)\cos\nu\pi - \mathsf{J}_{-\nu}(x)}{\sin\nu\pi}, & \text{for } x \geq 0 \text{ and non-integral } \nu \\ \lim\limits_{\mu\to\nu}\dfrac{\mathsf{J}_\mu(x)\cos\mu\pi - \mathsf{J}_{-\mu}(x)}{\sin\mu\pi}, & \text{for } x \geq 0 \text{ and integral } \nu \end{cases}.$$

3 *Note:* The effect of calling each of these functions is implementation-defined if `nu >= 128`.

4 See 6.2.8.

6.2.11 (incomplete) elliptic integral of the first kind [num.sf.ellF]

```
double       ellint_1(double k, double phi);
float        ellint_1f(float k, float phi);
long double  ellint_1l(long double k, long double phi);
```

1 *Effects:* These functions compute the incomplete elliptic integral of the first kind of their respective arguments `k` and `phi` (`phi` measured in radians).

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

2 *Returns:* The ellint_1 functions return

$$\mathsf{F}(k,\phi) = \int_0^{\phi} \frac{\mathsf{d}\theta}{\sqrt{1-k^2\sin^2\theta}}, \quad \text{for } |k| \le 1.$$

6.2.12 (incomplete) elliptic integral of the second kind [num.sf.ellE]

```
double       ellint_2(double k, double phi);
float        ellint_2f(float k, float phi);
long double  ellint_2l(long double k, long double phi);
```

1 *Effects:* These functions compute the incomplete elliptic integral of the second kind of their respective arguments k and phi (phi measured in radians).

2 *Returns:* The ellint_2 functions return

$$\mathsf{E}(k,\phi) = \int_0^{\phi} \sqrt{1-k^2\sin^2\theta}\,\mathsf{d}\theta, \quad \text{for } |k| \le 1.$$

6.2.13 (incomplete) elliptic integral of the third kind [num.sf.ellP]

```
double       ellint_3(double k, double nu, double phi);
float        ellint_3f(float k, float nu, float phi);
long double  ellint_3l(long double k, long double nu, long double phi);
```

1 *Effects:* These functions compute the incomplete elliptic integral of the third kind of their respective arguments k, nu, and phi (phi measured in radians).

2 *Returns:* The ellint_3 functions return

$$\Pi(\nu,k,\phi) = \int_0^{\phi} \frac{\mathsf{d}\theta}{(1-\nu\sin^2\theta)\sqrt{1-k^2\sin^2\theta}}, \quad \text{for } |k| \le 1.$$

6.2.14 exponential integral [num.sf.ei]

```
double       expint(double x);
float        expintf(float x);
long double  expintl(long double x);
```

1 *Effects:* These functions compute the exponential integral of their respective arguments x.

2 *Returns:* The expint functions return

$$\mathsf{Ei}(x) = -\int_{-x}^{\infty} \frac{e^{-t}}{t}\,\mathsf{d}t\ .$$

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

6.2.15 Hermite polynomials [num.sf.Hn]

```
double       hermite(unsigned n, double x);
float        hermitef(unsigned n, float x);
long double  hermitel(unsigned n, long double x);
```

1 *Effects:* These functions compute the Hermite polynomials of their respective arguments `n` and `x`.

2 *Returns:* The `hermite` functions return

$$\mathsf{H}_n(x) = (-1)^n e^{x^2} \frac{\mathsf{d}^n}{\mathsf{d}x^n} e^{-x^2} .$$

3 *Note:* The effect of calling each of these functions is implementation-defined if `n >= 128`.

6.2.16 Laguerre polynomials [num.sf.Ln]

```
double       laguerre(unsigned n, double x);
float        laguerref(unsigned n, float x);
long double  laguerrel(unsigned n, long double x);
```

1 *Effects:* These functions compute the Laguerre polynomials of their respective arguments `n` and `x`.

2 *Returns:* The `laguerre` functions return

$$\mathsf{L}_n(x) = \frac{e^x}{n!} \frac{\mathsf{d}^n}{\mathsf{d}x^n} (x^n e^{-x}), \quad \text{for } x \geq 0.$$

3 *Note:* The effect of calling each of these functions is implementation-defined if `n >= 128`.

6.2.17 Legendre polynomials [num.sf.Pl]

```
double       legendre(unsigned l, double x);
float        legendref(unsigned l, float x);
long double  legendrel(unsigned l, long double x);
```

1 *Effects:* These functions compute the Legendre polynomials of their respective arguments `l` and `x`.

2 *Returns:* The `legendre` functions return

$$\mathsf{P}_\ell(x) = \frac{1}{2^\ell \ell!} \frac{\mathsf{d}^\ell}{\mathsf{d}x^\ell} (x^2 - 1)^\ell, \quad \text{for } |x| \leq 1.$$

3 *Note:* The effect of calling each of these functions is implementation-defined if `l >= 128`.

--`,,```,,,,````-`-`,,`,,`,`,,`---

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

6.2.18 Riemann zeta function [num.sf.riemannzeta]

```
double       riemann_zeta(double x);
float        riemann_zetaf(float x);
long double  riemann_zetal(long double x);
```

1 *Effects:* These functions compute the Riemann zeta function of their respective arguments `x`.

2 *Returns:* The `riemann_zeta` functions return

$$\zeta(x) = \begin{cases} \sum_{k=1}^{\infty} k^{-x}, & \text{for } x > 1 \\ \frac{1}{1-2^{1-x}} \sum_{k=1}^{\infty} (-1)^{k-1} k^{-x}, & \text{for } 0 \le x \le 1 \\ 2^x \pi^{x-1} \sin(\frac{\pi x}{2}) \Gamma(1-x)\, \zeta(1-x), & \text{for } x < 0 \end{cases} .$$

6.2.19 spherical Bessel functions (of the first kind) [num.sf.j]

```
double       sph_bessel(unsigned n, double x);
float        sph_besself(unsigned n, float  x);
long double  sph_bessell(unsigned n, long double x);
```

1 *Effects:* These functions compute the spherical Bessel functions of the first kind of their respective arguments `n` and `x`.

2 *Returns:* The `sph_bessel` functions return

$$\mathsf{j}_n(x) = (\pi/2x)^{1/2} \mathsf{J}_{n+1/2}(x), \quad \text{for } x \ge 0.$$

3 *Note:* The effect of calling each of these functions is implementation-defined if `n >= 128`.

4 See 6.2.8.

6.2.20 spherical associated Legendre functions [num.sf.Ylm]

```
double       sph_legendre(unsigned l, unsigned m, double theta);
float        sph_legendref(unsigned l, unsigned m, float theta);
long double  sph_legendrel(unsigned l, unsigned m, long double theta);
```

1 *Effects:* These functions compute the spherical associated Legendre functions of their respective arguments `l`, `m`, and `theta` (`theta` measured in radians).

2 *Returns:* The `sph_legendre` functions return

$$\mathsf{Y}_\ell^m(\theta, 0)$$

where

$$\mathsf{Y}_\ell^m(\theta,\phi) = (-1)^m \left[\frac{(2\ell+1)}{4\pi} \frac{(\ell-m)!}{(\ell+m)!} \right]^{1/2} \mathsf{P}_\ell^m(\cos\theta) e^{im\phi}, \quad \text{for } |m| \le \ell.$$

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

3 *Note:* The effect of calling each of these functions is implementation-defined if `l >= 128`.

4 See 6.2.8.

6.2.21 spherical Neumann functions [num.sf.n]

```
double       sph_neumann(unsigned n, double x);
float        sph_neumannf(unsigned n, float  x);
long double  sph_neumannl(unsigned n, long double x);
```

1 *Effects:* These functions compute the spherical Neumann functions, also known as the spherical Bessel functions of the second kind, of their respective arguments `n` and `x`.

2 *Returns:* The `sph_neumann` functions return

$$\mathsf{n}_n(x) = (\pi/2x)^{1/2}\mathsf{N}_{n+1/2}(x), \quad \text{for } x \geq 0.$$

3 *Note:* The effect of calling each of these functions is implementation-defined if `n >= 128`.

4 See 6.2.10.

6.3 Additions to header <tgmath.h> [sf.tgmath]

1 The header <tgmath.h> includes the header <math.h> and defines the type-generic macros shown in Table 3.

2 Of the functions added by this document to <math.h> without an `f` (float) or `l` (long double) suffix, several have one or more parameters whose corresponding real type is `double`. For each such function there is a corresponding type-generic macro. [4] The parameters whose corresponding real type is `double` in the function synopsis are generic parameters. Use of the macro invokes a function whose corresponding real type and type domain are determined by the arguments for the generic parameters. [5]

3 Use of the macro invokes a function whose generic parameters have the corresponding real type determined as follows:

— First, if any argument for generic parameters has type `long double`, the type determined is `long double`.

— Otherwise, if any argument for generic parameters has type `double` or is of integer type, the type determined is `double`.

— Otherwise, the type determined is `float`.

4 For each unsuffixed function added to <math.h> the corresponding type-generic macro has the same name as the function. These type-generic macros are shown in Table 3.

[4] Like other function-like macros in Standard libraries, each *type-generic macro* can be suppressed to make available the corresponding ordinary function.

[5] If the type of the argument is not compatible with the type of the parameter for the selected function, the behavior is undefined.

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Table 3: Additions to header <tgmath.h> synopsis

Macros:		
assoc_laguerre	cyl_bessel_j	hermite
assoc_legendre	cyl_bessel_k	legendre
beta	cyl_neumann	laguerre
comp_ellint_1	ellint_1	riemann_zeta
comp_ellint_2	ellint_2	sph_bessel
comp_ellint_3	ellint_3	sph_legendre
cyl_bessel_i	expint	sph_neumann

5 If all arguments for generic parameters are real, then use of the macro invokes a real function; otherwise, use of the macro results in undefined behavior.

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Bibliography

[1] ISO/IEC TR 19768, *Information technology — Programming languages — Technical Report on C++ Library Extensions*

[2] ISO 31-11, *Quantities and units — Part 11: Mathematical signs and symbols for use in the physical sciences and technology*

[3] IEC 60559:1989, *Binary floating-point arithmetic for microprocessor systems*

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

(Blank page)

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Index

__STDC_MATH_SPEC_FUNCS__ , 9
__STDC_WANT_MATH_SPEC_FUNCS__ , 11

assoc_laguerre, 14
assoc_legendre, 14

Bessel functions, 16, 17, 20, 21
beta, 15

comp_ellint_1, 15
comp_ellint_2, 15
comp_ellint_3, 16
cyl_bessel_i, 16
cyl_bessel_j, 16
cyl_bessel_k, 17
cyl_neumann, 17

ellint_1, 17
ellint_2, 18
ellint_3, 18
elliptic integrals, 15–18
    complete, 15, 16
    incomplete, 17, 18
Eulerian integral of the first kind, *see* beta
expint, 18
exponential integral, 18

hermite, 19
H_n polynomials, 19

I_ν, 16

J_ν, 16
j_n, 20

K_ν, 17

laguerre, 19
Laguerre polynomials, 14
legendre, 19
Legendre functions, 19, 20
Legendre polynomials, 14

<math.h>, 11

N_ν, 17
NaN, 14
Neumann functions, 17, 21
n_n function, 21

P_l polynomials, 19
P_{lm}, 14

riemann_zeta, 20

special functions, 11–22
sph_bessel, 20
sph_legendre, 20
sph_neumann, 21
spherical harmonics, 20
standard headers, 11

<tgmath.h>, 21
type-generic macro, 21

undefined behavior, 22

$Y_l^m(\theta, \phi)$, 20

ζ function, 20

© ISO/IEC 2009 – All rights reserved

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

ICS 35.060

Price based on 19 pages

© ISO/IEC 2009 – All rights reserved
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale