

new/usr/src/head/fenv.h

1

```
*****
5193 Tue Apr 30 23:28:34 2019
new/usr/src/head/fenv.h
10882 math headers should stop supporting K&R C
*****
1 /*
2  * CDDL HEADER START
3  *
4  * The contents of this file are subject to the terms of the
5  * Common Development and Distribution License (the "License").
6  * You may not use this file except in compliance with the License.
7  *
8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9  * or http://www.opensolaris.org/os/licensing.
10 * See the License for the specific language governing permissions
11 * and limitations under the License.
12 *
13 * When distributing Covered Code, include this CDDL HEADER in each
14 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 * If applicable, add the following below this CDDL HEADER, with the
16 * fields enclosed by brackets "[]" replaced with your own identifying
17 * information: Portions Copyright [yyyy] [name of copyright owner]
18 *
19 * CDDL HEADER END
20 */
21 /*
22 * Copyright 2011 Nexenta Systems, Inc. All rights reserved.
23 */
24 /*
25 * Copyright 2004 Sun Microsystems, Inc. All rights reserved.
26 * Use is subject to license terms.
27 */

29 #ifndef _FENV_H
30 #define _FENV_H

32 #include <sys/feature_tests.h>

34 #ifdef __cplusplus
35 extern "C" {
36 #endif

38 #ifndef __P
39 #ifdef __STDC__
40 #define __P(p) p
41 #else
42 #define __P(p) ()
43 #endif
44 #endif /* !defined(__P) */

38 /*
39  * Rounding modes
40  */
41 #if defined(__sparc)

43 #define FE_TONEAREST 0
44 #define FE_TOWARDZERO 1
45 #define FE_UPWARD 2
46 #define FE_DOWNWARD 3

48 #elif defined(__i386) || defined(__amd64)

50 #define FE_TONEAREST 0
51 #define FE_DOWNWARD 1
52 #define FE_UPWARD 2
53 #define FE_TOWARDZERO 3
```

new/usr/src/head/fenv.h

2

```
55 #endif

57 extern int fegetround(void);
58 extern int fesetround(int);
65 extern int fegetround __P((void));
66 extern int fesetround __P((int));

60 #if (defined(__i386) || defined(__amd64)) && \
61     (!defined(_STRICT_STDC) || defined(__EXTENSIONS__))

63 #define FE_FLTPREC 0
64 #define FE_DBLPREC 2
65 #define FE_LDBLPREC 3

67 extern int fegetprec(void);
68 extern int fesetprec(int);
75 extern int fegetprec __P((void));
76 extern int fesetprec __P((int));

70 #endif

72 /*
73  * Exception flags
74  */
75 #if defined(__sparc)

77 #define FE_INEXACT 0x01
78 #define FE_DIVBYZERO 0x02
79 #define FE_UNDERFLOW 0x04
80 #define FE_OVERFLOW 0x08
81 #define FE_INVALID 0x10
82 #define FE_ALL_EXCEPT 0x1f

84 #elif defined(__i386) || defined(__amd64)

86 #define FE_INVALID 0x01
87 #define FE_DIVBYZERO 0x04
88 #define FE_OVERFLOW 0x08
89 #define FE_UNDERFLOW 0x10
90 #define FE_INEXACT 0x20
91 #define FE_ALL_EXCEPT 0x3d

93 #endif

95 typedef int fexcept_t;

97 extern int feclearexcept(int);
98 extern int feraiseexcept(int);
99 extern int fetestexcept(int);
100 extern int fegetexceptflag(fexcept_t *, int);
101 extern int fesetexceptflag(const fexcept_t *, int);
105 extern int feclearexcept __P((int));
106 extern int feraiseexcept __P((int));
107 extern int fetestexcept __P((int));
108 extern int fegetexceptflag __P((fexcept_t *, int));
109 extern int fesetexceptflag __P((const fexcept_t *, int));

103 #if !defined(_STRICT_STDC) || defined(__EXTENSIONS__)

105 /*
106  * Exception handling extensions
107  */
108 #define FEX_NOHANDLER -1
109 #define FEX_NONSTOP 0
110 #define FEX_ABORT 1
```

new/usr/src/head/fenv.h

3

```

111 #define FEX_SIGNAL      2
112 #define FEX_CUSTOM      3

114 #define FEX_INEXACT     0x001
115 #define FEX_DIVBYZERO   0x002
116 #define FEX_UNDERFLOW  0x004
117 #define FEX_OVERFLOW    0x008
118 #define FEX_INV_ZDZ     0x010
119 #define FEX_INV_IDI     0x020
120 #define FEX_INV_ISI     0x040
121 #define FEX_INV_ZMI     0x080
122 #define FEX_INV_SQRT    0x100
123 #define FEX_INV_SNaN    0x200
124 #define FEX_INV_INT     0x400
125 #define FEX_INV_CMP     0x800
126 #define FEX_INVALID     0xff0
127 #define FEX_COMMON      (FEX_INVALID | FEX_DIVBYZERO | FEX_OVERFLOW)
128 #define FEX_ALL         (FEX_COMMON | FEX_UNDERFLOW | FEX_INEXACT)
129 #define FEX_NONE        0

131 #define FEX_NUM_EXC     12

133 /* structure to hold a numeric value in any format used by the FPU */
134 typedef struct {
135     enum fex_nt {
136         fex_nodata    = 0,
137         fex_int        = 1,
138         fex_llong      = 2,
139         fex_float      = 3,
140         fex_double     = 4,
141         fex_ldouble    = 5
142     } type;
143     union {
144         int            i;
145     } #if !defined(_STRICT_STDC) && !defined(_NO_LONGLONG) || defined(_STDC_C99) || \
146         defined(__C99FEATURES__)
147         long long      l;
148     #else
149         struct {
150             int        l[2];
151         } l;
152     #endif
153     float              f;
154     double             d;
155     long double       q;
156 } val;
157 } fex_numeric_t;
    unchanged portion omitted

180 extern int fex_get_handling(int);
181 extern int fex_set_handling(int, int, void (*)());
182 extern int fex_get_handling __P((int));
183 extern int fex_set_handling __P((int, int, void (*)()));

183 extern void fex_getexcepthandler(fex_handler_t *, int);
184 extern void fex_setexcepthandler(const fex_handler_t *, int);
191 extern void fex_getexcepthandler __P((fex_handler_t *, int));
192 extern void fex_setexcepthandler __P((const fex_handler_t *, int));

186 #ifdef __STDC__
187 #include <stdio_tag.h>
188 #ifndef _FILEDEFED
189 #define _FILEDEFED
190 typedef __FILE FILE;
191 #endif
192 #endif

```

new/usr/src/head/fenv.h

4

```

193 extern FILE *fex_get_log(void);
194 extern int fex_set_log(FILE *);
195 extern int fex_get_log_depth(void);
196 extern int fex_set_log_depth(int);
197 extern void fex_log_entry(const char *);
201 extern FILE *fex_get_log __P((void));
202 extern int fex_set_log __P((FILE *));
203 extern int fex_get_log_depth __P((void));
204 extern int fex_set_log_depth __P((int));
205 extern void fex_log_entry __P((const char *));

199 #define __fex_handler_t fex_handler_t

201 #else

203 typedef struct {
204     int            __mode;
205     void          (*__handler)();
206 } __fex_handler_t[12];
    unchanged portion omitted

218 #ifdef __STDC__
219 extern const fenv_t __fenv_dfl_env;
220 #else
221 extern fenv_t __fenv_dfl_env;
222 #endif

224 #define FE_DFL_ENV      (&__fenv_dfl_env)

226 extern int fegetenv(fenv_t *);
227 extern int fesetenv(const fenv_t *);
228 extern int feholdexcept(fenv_t *);
229 extern int feupdateenv(const fenv_t *);
234 extern int fegetenv __P((fenv_t *));
235 extern int fesetenv __P((const fenv_t *));
236 extern int feholdexcept __P((fenv_t *));
237 extern int feupdateenv __P((const fenv_t *));

231 #if !defined(_STRICT_STDC) || defined(__EXTENSIONS__)
232 extern void fex_merge_flags(const fenv_t *);
240 extern void fex_merge_flags __P((const fenv_t *));
233 #endif

235 #ifdef __cplusplus
236 }
    unchanged portion omitted

```

new/usr/src/head/floatingpoint.h

1

```
*****
6482 Tue Apr 30 23:28:34 2019
new/usr/src/head/floatingpoint.h
10882 math headers should stop supporting K&R C
*****
1 /*
2  * CDDL HEADER START
3  *
4  * The contents of this file are subject to the terms of the
5  * Common Development and Distribution License (the "License").
6  * You may not use this file except in compliance with the License.
7  *
8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9  * or http://www.opensolaris.org/os/licensing.
10 * See the License for the specific language governing permissions
11 * and limitations under the License.
12 *
13 * When distributing Covered Code, include this CDDL HEADER in each
14 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 * If applicable, add the following below this CDDL HEADER, with the
16 * fields enclosed by brackets "[]" replaced with your own identifying
17 * information: Portions Copyright [yyyy] [name of copyright owner]
18 *
19 * CDDL HEADER END
20 */
21 /*      Copyright (C) 1989 AT&T */
22 /*      All Rights Reserved */

24 /*
25  * Copyright 2011 Nexenta Systems, Inc. All rights reserved.
26  */
27 /*
28  * Copyright 2004 Sun Microsystems, Inc. All rights reserved.
29  * Use is subject to license terms.
30  */

32 #ifndef _FLOATINGPOINT_H
33 #define _FLOATINGPOINT_H

35 #ifdef __STDC__
36 #include <stdio_tag.h>
37 #endif
38 #include <sys/ieeefp.h>

40 #ifdef __cplusplus
41 extern "C" {
42 #endif

44 /*
45  * <floatingpoint.h> contains definitions for constants, types, variables,
46  * and functions for:
47  *     IEEE floating-point arithmetic base conversion;
48  *     IEEE floating-point arithmetic modes;
49  *     IEEE floating-point arithmetic exception handling.
50  */

52 #ifndef __P
53 #ifdef __STDC__
54 #define __P(p) p
55 #else
56 #define __P(p) ()
57 #endif
58 #endif /* !defined(__P) */

51 #if defined(__STDC__) && !defined(_FILEDEFED)
52 #define _FILEDEFED
```

new/usr/src/head/floatingpoint.h

2

```
53 typedef __FILE FILE;
54 #endif

56 typedef int sigfpe_code_type; /* Type of SIGFPE code. */

58 typedef void (*sigfpe_handler_type)(); /* Pointer to exception handler */

60 #define SIGFPE_DEFAULT (void (*)())0 /* default exception handling */
61 #define SIGFPE_IGNORE (void (*)())1 /* ignore this exception or code */
62 #define SIGFPE_ABORT (void (*)())2 /* force abort on exception */

64 extern sigfpe_handler_type sigfpe(sigfpe_code_type, sigfpe_handler_type);
73 extern sigfpe_handler_type sigfpe __P((sigfpe_code_type, sigfpe_handler_type));

66 /*
67  * Types for IEEE floating point.
68  */
69 typedef float single;

71 #ifndef _EXTENDED
72 #define _EXTENDED
73 typedef unsigned extended[3];
74 #endif

76 typedef long double quadruple; /* Quadruple-precision type. */

78 typedef unsigned fp_exception_field_type;
79 /*
80  * A field containing fp_exceptions OR'ed
81  * together.
82  */
83 /*
84  * Definitions for base conversion.
85  */
86 #define DECIMAL_STRING_LENGTH 512 /* Size of buffer in decimal_record. */

88 typedef char decimal_string[DECIMAL_STRING_LENGTH];
89 /* Decimal significand. */

91 typedef struct {
92     enum fp_class_type fpclass;
93     int sign;
94     int exponent;
95     decimal_string ds; /* Significand - each char contains an ascii */
96 /* digit, except the string-terminating */
97 /* ascii null. */
98     int more; /* On conversion from decimal to binary, != 0 */
99 /* indicates more non-zero digits following */
100 /* ds. */
101     int ndigits; /* On fixed_form conversion from binary to */
102 /* decimal, contains number of digits */
103 /* required for ds. */
104 } decimal_record;
    unchanged_portion_omitted

140 extern void single_to_decimal(single *, decimal_mode *, decimal_record *,
141     fp_exception_field_type *);
142 extern void double_to_decimal(double *, decimal_mode *, decimal_record *,
143     fp_exception_field_type *);
144 extern void extended_to_decimal(extended *, decimal_mode *,
145     decimal_record *, fp_exception_field_type *);
146 extern void quadruple_to_decimal(quadruple *, decimal_mode *,
147     decimal_record *, fp_exception_field_type *);

149 extern void decimal_to_single(single *, decimal_mode *, decimal_record *,
150     fp_exception_field_type *);
```

```

151 extern void decimal_to_double(double *, decimal_mode *, decimal_record *,
152     fp_exception_field_type *);
153 extern void decimal_to_extended(extended *, decimal_mode *,
154     decimal_record *, fp_exception_field_type *);
155 extern void decimal_to_quadruple(quadruple *, decimal_mode *,
156     decimal_record *, fp_exception_field_type *);

158 extern void string_to_decimal(char **, int, int, decimal_record *,
159     enum decimal_string_form *, char **);
160 extern void func_to_decimal(char **, int, int, decimal_record *,
161     enum decimal_string_form *, char **);
162 extern void single_to_decimal __P((single *, decimal_mode *, decimal_record *,
163     fp_exception_field_type *));
164 extern void double_to_decimal __P((double *, decimal_mode *, decimal_record *,
165     fp_exception_field_type *));
166 extern void extended_to_decimal __P((extended *, decimal_mode *,
167     decimal_record *, fp_exception_field_type *));
168 extern void quadruple_to_decimal __P((quadruple *, decimal_mode *,
169     decimal_record *, fp_exception_field_type *));

171 extern void decimal_to_single __P((single *, decimal_mode *, decimal_record *,
172     fp_exception_field_type *));
173 extern void decimal_to_double __P((double *, decimal_mode *, decimal_record *,
174     fp_exception_field_type *));
175 extern void decimal_to_extended __P((extended *, decimal_mode *,
176     decimal_record *, fp_exception_field_type *));
177 extern void decimal_to_quadruple __P((quadruple *, decimal_mode *,
178     decimal_record *, fp_exception_field_type *));

180 extern void string_to_decimal __P((char **, int, int, decimal_record *,
181     enum decimal_string_form *, char **);
182 extern void func_to_decimal __P((char **, int, int, decimal_record *,
183     enum decimal_string_form *, char **);
184 extern void file_to_decimal(char **, int, int, decimal_record *,
185     int (*)(void), int *, int (*)(int));
186 extern void file_to_decimal __P((char **, int, int, decimal_record *,
187     enum decimal_string_form *, char **,
188     FILE *, int *);
189

191 extern char *seconvert(single *, int, int *, int *, char *);
192 extern char *sfconvert(single *, int, int *, int *, char *);
193 extern char *sgconvert(single *, int, int, char *);
194 extern char *econvert(double, int, int *, int *, char *);
195 extern char *fconvert(double, int, int *, int *, char *);
196 extern char *gconvert(double, int, int, char *);
197 extern char *qeconvert(quadruple *, int, int *, int *, char *);
198 extern char *qfconvert(quadruple *, int, int *, int *, char *);
199 extern char *qgconvert(quadruple *, int, int, char *);

201 extern char *ecvt(double, int, int *, int *);
202 extern char *fcvt(double, int, int *, int *);
203 extern char *gcvt(double, int, char *);
204 extern char *seconvert __P((single *, int, int *, int *, char *));
205 extern char *sfconvert __P((single *, int, int *, int *, char *));
206 extern char *sgconvert __P((single *, int, int, char *));
207 extern char *econvert __P((double, int, int *, int *, char *));
208 extern char *fconvert __P((double, int, int *, int *, char *));
209 extern char *gconvert __P((double, int, int, char *));
210 extern char *qeconvert __P((quadruple *, int, int *, int *, char *));
211 extern char *qfconvert __P((quadruple *, int, int *, int *, char *));
212 extern char *qgconvert __P((quadruple *, int, int, char *));

214 extern char *ecvt __P((double, int, int *, int *));
215 extern char *fcvt __P((double, int, int *, int *));
216 extern char *gcvt __P((double, int, char *));

```

```

181 #if __cplusplus >= 199711L
182 namespace std {
183 #endif
184 /*
185  * ANSI C Standard says the following entry points should be
186  * prototyped in <stdlib.h>. They are now, but weren't before.
187  */
188 extern double atof(const char *);
189 extern double strtod(const char *, char **);
190 extern double atof __P((const char *));
191 extern double strtod __P((const char *, char **));
192 #if __cplusplus >= 199711L
193 }
194 #endif
195 _____ unchanged_portion_omitted _____

```

new/usr/src/head/iso/math_c99.h

1

```
*****
18816 Tue Apr 30 23:28:35 2019
new/usr/src/head/iso/math_c99.h
10882 math headers should stop supporting K&R C
*****
1 /*
2  * CDDL HEADER START
3  *
4  * The contents of this file are subject to the terms of the
5  * Common Development and Distribution License (the "License").
6  * You may not use this file except in compliance with the License.
7  *
8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9  * or http://www.opensolaris.org/os/licensing.
10 * See the License for the specific language governing permissions
11 * and limitations under the License.
12 *
13 * When distributing Covered Code, include this CDDL HEADER in each
14 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 * If applicable, add the following below this CDDL HEADER, with the
16 * fields enclosed by brackets "[]" replaced with your own identifying
17 * information: Portions Copyright [yyyy] [name of copyright owner]
18 *
19 * CDDL HEADER END
20 */
21 /*
22  * Copyright 2011 Nexenta Systems, Inc. All rights reserved.
23 */
24 /*
25  * Copyright 2005 Sun Microsystems, Inc. All rights reserved.
26  * Use is subject to license terms.
27 */

29 #ifndef _ISO_MATH_C99_H
30 #define _ISO_MATH_C99_H

32 #include <sys/isa_defs.h>
33 #include <sys/feature_tests.h>

35 #ifdef __cplusplus
36 extern "C" {
37 #endif

39 #undef FP_ZERO
40 #define FP_ZERO 0
41 #undef FP_SUBNORMAL
42 #define FP_SUBNORMAL 1
43 #undef FP_NORMAL
44 #define FP_NORMAL 2
45 #undef FP_INFINITY
46 #define FP_INFINITY 3
47 #undef FP_NAN
48 #define FP_NAN 4
49 #ifndef __P
50 #ifdef __STDC__
51 #define __P(p) p
52 #else
53 #define __P(p) ()
54 #endif
55 #endif /* !defined(__P) */

50 #if defined(__STDC_C99) || _XOPEN_SOURCE - 0 >= 600 || defined(__C99FEATURES__)
51 #if defined(__GNUC__)
52 #undef HUGE_VAL
53 #define HUGE_VAL (__builtin_huge_val())
54 #undef HUGE_VALF
```

new/usr/src/head/iso/math_c99.h

2

```
55 #define HUGE_VALF (__builtin_huge_valf())
56 #undef HUGE_VALL
57 #define HUGE_VALL (__builtin_huge_vall())
58 #undef INFINITY
59 #define INFINITY (__builtin_inff())
60 #undef NAN
61 #define NAN (__builtin_nanf(""))

63 /*
64  * C99 7.12.3 classification macros
65 */
66 #undef isnan
67 #undef isinf
68 #if __GNUC__ >= 4
69 #define isnan(x) __builtin_isnan(x)
70 #define isinf(x) __builtin_isinf(x)
71 #define fpclassify(x) __builtin_fpclassify(FP_NAN, FP_INFINITY, FP_NORMAL, \
72 FP_SUBNORMAL, FP_ZERO, x)
73 #define isfinite(x) __builtin_isfinite(x)
74 #define isnormal(x) __builtin_isnormal(x)
75 #define signbit(x) __builtin_signbit(x)
76 #else /* __GNUC__ >= 4 */
77 #define isnan(x) __extension__( \
78 { __typeof(x) __x_n = (x); \
79 __builtin_isunordered(__x_n, __x_n); })
80 #define isinf(x) __extension__( \
81 { __typeof(x) __x_i = (x); \
82 __x_i == (__typeof(__x_i)) INFINITY || \
83 __x_i == (__typeof(__x_i)) (-INFINITY); })
84 #endif
85 #define isfinite(x) __extension__( \
86 { __typeof(x) __x_f = (x); \
87 !isnan(__x_f) && !isinf(__x_f); })
88 #undef isnormal
89 #define isnormal(x) __extension__( \
90 { __typeof(x) __x_r = (x); isfinite(__x_r) && \
91 (sizeof (__x_r) == sizeof (float) ? \
92 __builtin_fabsf(__x_r) >= _FLT_MIN : \
93 sizeof (__x_r) == sizeof (double) ? \
94 __builtin_fabs(__x_r) >= _DBL_MIN : \
95 __builtin_fabsl(__x_r) >= _LDBL_MIN); })
96 #undef fpclassify
97 #define fpclassify(x) __extension__( \
98 { __typeof(x) __x_c = (x); \
99 isnan(__x_c) ? FP_NAN : \
100 isinf(__x_c) ? FP_INFINITY : \
101 isnormal(__x_c) ? FP_NORMAL : \
102 __x_c == (__typeof(__x_c)) 0 ? FP_ZERO : \
103 FP_SUBNORMAL; })
104 #undef signbit
105 #if defined(_BIG_ENDIAN)
106 #define signbit(x) __extension__( \
107 { __typeof(x) __x_s = (x); \
108 (int)((unsigned *)&__x_s >> 31); })
109 #elif defined(_LITTLE_ENDIAN)
110 #define signbit(x) __extension__( \
111 { __typeof(x) __x_s = (x); \
112 (sizeof (__x_s) == sizeof (float) ? \
113 (int)((unsigned *)&__x_s >> 31) : \
114 (int)((unsigned *)&__x_s >> 31) : \
115 sizeof (__x_s) == sizeof (double) ? \
116 (int)((unsigned *)&__x_s[1] >> 31) : \
117 (int)((unsigned short *)&__x_s[4] >> 15); })
```

```

117 #endif /* defined(__BIG_ENDIAN) */
118 #endif /* __GNUC__ >= 4 */
119         (int) (((unsigned *) &__x_s)[1] >> 31) : \
120         (int) (((unsigned short *) &__x_s)[4] >> 15)); })
121 #endif

122 /*
123 * C99 7.12.14 comparison macros
124 */
125 #undef isgreater
126 #define isgreater(x, y)      __builtin_isgreater(x, y)
127 #undef isgreaterequal
128 #define isgreaterequal(x, y) __builtin_isgreaterequal(x, y)
129 #undef isless
130 #define isless(x, y)        __builtin_isless(x, y)
131 #undef islessequal
132 #define islessequal(x, y)  __builtin_islessequal(x, y)
133 #undef islessgreater
134 #define islessgreater(x, y) __builtin_islessgreater(x, y)
135 #undef isunordered
136 #define isunordered(x, y)  __builtin_isunordered(x, y)
137 #else /* defined(__GNUC__) */
138 #undef HUGE_VAL
139 #define HUGE_VAL      __builtin_huge_val
140 #undef HUGE_VALF
141 #define HUGE_VALF    __builtin_huge_valf
142 #undef HUGE_VALL
143 #define HUGE_VALL    __builtin_huge_vall
144 #undef INFINITY
145 #define INFINITY     __builtin_infinity
146 #undef NAN
147 #define NAN          __builtin_nan

148 /*
149 * C99 7.12.3 classification macros
150 */
151 #undef fpclassify
152 #define fpclassify(x)  __builtin_fpclassify(x)
153 #undef isfinite
154 #define isfinite(x)   __builtin_isfinite(x)
155 #undef isinf
156 #define isinf(x)     __builtin_isinf(x)
157 #undef isnan
158 #define isnan(x)     __builtin_isnan(x)
159 #undef isnormal
160 #define isnormal(x)  __builtin_isnormal(x)
161 #undef signbit
162 #define signbit(x)   __builtin_signbit(x)

163 /*
164 * C99 7.12.14 comparison macros
165 */
166 #undef isgreater
167 #define isgreater(x, y)      ((x) __builtin_isgreater(y))
168 #undef isgreaterequal
169 #define isgreaterequal(x, y) ((x) __builtin_isgreaterequal(y))
170 #undef isless
171 #define isless(x, y)        ((x) __builtin_isless(y))
172 #undef islessequal
173 #define islessequal(x, y)  ((x) __builtin_islessequal(y))
174 #undef islessgreater
175 #define islessgreater(x, y) ((x) __builtin_islessgreater(y))
176 #undef isunordered
177 #define isunordered(x, y)  ((x) __builtin_isunordered(y))
178 #endif /* defined(__GNUC__) */
179 #endif /* defined(__STDC_C99) || __XOPEN_SOURCE - 0 >= 600 || ... */

```

```

181 #if defined(__EXTENSIONS__) || defined(__STDC_C99) || \
182     (defined(__STRICT_STDC) && !defined(__XOPEN_OR_POSIX)) || \
183     defined(__C99FEATURES__)
184 #if defined(__FLT_EVAL_METHOD__) && __FLT_EVAL_METHOD__ - 0 == 0
185 typedef float float_t;
186 typedef double double_t;
187 #elif __FLT_EVAL_METHOD__ - 0 == 1
188 typedef double float_t;
189 typedef double double_t;
190 #elif __FLT_EVAL_METHOD__ - 0 == 2
191 typedef long double float_t;
192 typedef long double double_t;
193 #elif defined(__sparc) || defined(__amd64)
194 typedef float float_t;
195 typedef double double_t;
196 #elif defined(__i386)
197 typedef long double float_t;
198 typedef long double double_t;
199 #endif

200 #undef FP_ZERO
201 #define FP_ZERO      0
202 #undef FP_SUBNORMAL
203 #define FP_SUBNORMAL 1
204 #undef FP_NORMAL
205 #define FP_NORMAL    2
206 #undef FP_INFINITE
207 #define FP_INFINITE  3
208 #undef FP_NAN
209 #define FP_NAN       4

210 #undef FP_ILOGB0
211 #define FP_ILOGB0    (-2147483647)
212 #undef FP_ILOGBNAN
213 #define FP_ILOGBNAN  2147483647

214 #undef MATH_ERRNO
215 #define MATH_ERRNO   1
216 #undef MATH_ERREXCEPT
217 #define MATH_ERREXCEPT 2
218 #undef math_errhandling
219 #define math_errhandling MATH_ERREXCEPT

220 extern double acosh(double);
221 extern double asinh(double);
222 extern double atanh(double);

223 extern double exp2(double);
224 extern double expm1(double);
225 extern int ilogb(double);
226 extern double log1p(double);
227 extern double log2(double);
228 extern double logb(double);
229 extern double scalbn(double, int);
230 extern double scalbln(double, long int);

231 extern double cbrt(double);
232 extern double hypot(double, double);

233 extern double erf(double);
234 extern double erfc(double);
235 extern double lgamma(double);
236 extern double tgamma(double);

237 extern double nearbyint(double);

```

```

235 extern double rint(double);
236 extern long int lrint(double);
237 extern double round(double);
238 extern long int lround(double);
239 extern double trunc(double);

241 extern double remainder(double, double);
242 extern double remquo(double, double, int *);

244 extern double copysign(double, double);
245 extern double nan(const char *);
246 extern double nextafter(double, double);
247 extern double nexttoward(double, long double);

249 extern double fdim(double, double);
250 extern double fmax(double, double);
251 extern double fmin(double, double);

253 extern double fma(double, double, double);

255 extern float acosf(float);
256 extern float asinf(float);
257 extern float atanf(float);
258 extern float atan2f(float, float);
259 extern float cosf(float);
260 extern float sinf(float);
261 extern float tanf(float);

263 extern float acoshf(float);
264 extern float asinhf(float);
265 extern float atanhf(float);
266 extern float coshf(float);
267 extern float sinhlf(float);
268 extern float tanhf(float);

270 extern float expf(float);
271 extern float exp2f(float);
272 extern float expm1f(float);
273 extern float frexpf(float, int *);
274 extern int ilogbf(float);
275 extern float ldexpf(float, int);
276 extern float logf(float);
277 extern float log10f(float);
278 extern float log1pf(float);
279 extern float log2f(float);
280 extern float logbf(float);
281 extern float modff(float, float *);
282 extern float scalbnf(float, int);
283 extern float scalblnf(float, long int);

285 extern float cbrtf(float);
286 extern float fabsf(float);
287 extern float hypotf(float, float);
288 extern float powf(float, float);
289 extern float sqrtf(float);

291 extern float erff(float);
292 extern float erfcf(float);
293 extern float lgammaf(float);
294 extern float tgammaf(float);

296 extern float ceilf(float);
297 extern float floorf(float);
298 extern float nearbyintf(float);
299 extern float rintf(float);
300 extern long int lrintf(float);

```

```

301 extern float roundf(float);
302 extern long int lroundf(float);
303 extern float truncf(float);

305 extern float fmodf(float, float);
306 extern float remainderf(float, float);
307 extern float remquo(float, float, int *);

309 extern float copysignf(float, float);
310 extern float nanf(const char *);
311 extern float nextafterf(float, float);
312 extern float nexttowardf(float, long double);

314 extern float fdimf(float, float);
315 extern float fmaxf(float, float);
316 extern float fminf(float, float);

318 extern float fmaf(float, float, float);

320 extern long double acosl(long double);
321 extern long double asinl(long double);
322 extern long double atanl(long double);
323 extern long double atan2l(long double, long double);
324 extern long double cosl(long double);
325 extern long double sinl(long double);
326 extern long double tanl(long double);

328 extern long double acoshl(long double);
329 extern long double asinhl(long double);
330 extern long double atanhlf(long double);
331 extern long double coshl(long double);
332 extern long double sinhl(long double);
333 extern long double tanhl(long double);

335 extern long double expl(long double);
336 extern long double exp2l(long double);
337 extern long double expm1l(long double);
338 extern long double frexpl(long double, int *);
339 extern int ilogbl(long double);
340 extern long double ldexpl(long double, int);
341 extern long double logl(long double);
342 extern long double log10l(long double);
343 extern long double log1pl(long double);
344 extern long double log2l(long double);
345 extern long double logbl(long double);
346 extern long double modfl(long double, long double *);
347 extern long double scalbnl(long double, int);
348 extern long double scalblnl(long double, long int);

350 extern long double cbrtl(long double);
351 extern long double fabsl(long double);
352 extern long double hypotl(long double, long double);
353 extern long double powl(long double, long double);
354 extern long double sqrtl(long double);

356 extern long double erfl(long double);
357 extern long double erfcl(long double);
358 extern long double lgammal(long double);
359 extern long double tgamma(long double);

361 extern long double ceill(long double);
362 extern long double floo1l(long double);
363 extern long double nearbyintl(long double);
364 extern long double rintl(long double);
365 extern long int lrintl(long double);
366 extern long double roundl(long double);

```

```

367 extern long int lroundl(long double);
368 extern long double truncf(long double);

370 extern long double fmodl(long double, long double);
371 extern long double remainderl(long double, long double);
372 extern long double remquo1(long double, long double, int *);

374 extern long double copysignl(long double, long double);
375 extern long double nanl(const char *);
376 extern long double nextafterl(long double, long double);
377 extern long double nexttowardl(long double, long double);

379 extern long double fdiml(long double, long double);
380 extern long double fmaxl(long double, long double);
381 extern long double fminl(long double, long double);
216 extern double acosh __P((double));
217 extern double asinh __P((double));
218 extern double atanh __P((double));

220 extern double exp2 __P((double));
221 extern double expm1 __P((double));
222 extern int ilogb __P((double));
223 extern double log1p __P((double));
224 extern double log2 __P((double));
225 extern double logb __P((double));
226 extern double scalbn __P((double, int));
227 extern double scalbln __P((double, long int));

229 extern double cbrt __P((double));
230 extern double hypot __P((double, double));

232 extern double erf __P((double));
233 extern double erfc __P((double));
234 extern double lgamma __P((double));
235 extern double tgamma __P((double));

237 extern double nearbyint __P((double));
238 extern double rint __P((double));
239 extern long int lrint __P((double));
240 extern double round __P((double));
241 extern long int lround __P((double));
242 extern double trunc __P((double));

244 extern double remainder __P((double, double));
245 extern double remquo __P((double, double, int *));

247 extern double copysign __P((double, double));
248 extern double nan __P((const char *));
249 extern double nextafter __P((double, double));
250 extern double nexttoward __P((double, long double));

252 extern double fdim __P((double, double));
253 extern double fmax __P((double, double));
254 extern double fmin __P((double, double));

256 extern double fma __P((double, double, double));

258 extern float acosf __P((float));
259 extern float asinf __P((float));
260 extern float atanf __P((float));
261 extern float atan2f __P((float, float));
262 extern float cosf __P((float));
263 extern float sinf __P((float));
264 extern float tanf __P((float));

266 extern float acoshf __P((float));

```

```

267 extern float asinhf __P((float));
268 extern float atanhf __P((float));
269 extern float coshf __P((float));
270 extern float sinhf __P((float));
271 extern float tanhf __P((float));

273 extern float expf __P((float));
274 extern float exp2f __P((float));
275 extern float expm1f __P((float));
276 extern float frexpf __P((float, int *));
277 extern int ilogbf __P((float));
278 extern float ldexpf __P((float, int));
279 extern float logf __P((float));
280 extern float log10f __P((float));
281 extern float log1pf __P((float));
282 extern float log2f __P((float));
283 extern float logbf __P((float));
284 extern float modff __P((float, float *));
285 extern float scalbnf __P((float, int));
286 extern float scalblnf __P((float, long int));

288 extern float cbrtf __P((float));
289 extern float fabsf __P((float));
290 extern float hypotf __P((float, float));
291 extern float powf __P((float, float));
292 extern float sqrtf __P((float));

294 extern float erff __P((float));
295 extern float erfcf __P((float));
296 extern float lgammaf __P((float));
297 extern float tgammaf __P((float));

299 extern float ceilf __P((float));
300 extern float floorf __P((float));
301 extern float nearbyintf __P((float));
302 extern float rintf __P((float));
303 extern long int lrintf __P((float));
304 extern float roundf __P((float));
305 extern long int lroundf __P((float));
306 extern float truncf __P((float));

308 extern float fmodf __P((float, float));
309 extern float remainderf __P((float, float));
310 extern float remquof __P((float, float, int *));

312 extern float copysignf __P((float, float));
313 extern float nanf __P((const char *));
314 extern float nextafterf __P((float, float));
315 extern float nexttowardf __P((float, long double));

317 extern float fdimf __P((float, float));
318 extern float fmaxf __P((float, float));
319 extern float fminf __P((float, float));

321 extern float fmaf __P((float, float, float));

323 extern long double acosl __P((long double));
324 extern long double asinl __P((long double));
325 extern long double atanl __P((long double));
326 extern long double atan2l __P((long double, long double));
327 extern long double cosl __P((long double));
328 extern long double sinl __P((long double));
329 extern long double tanl __P((long double));

331 extern long double acoshl __P((long double));
332 extern long double asinhl __P((long double));

```



```

333 extern long double atanhl __P((long double));
334 extern long double coshl __P((long double));
335 extern long double sinhl __P((long double));
336 extern long double tanhl __P((long double));

338 extern long double expl __P((long double));
339 extern long double exp2l __P((long double));
340 extern long double expml __P((long double));
341 extern long double frexpl __P((long double, int *));
342 extern int ilogbl __P((long double));
343 extern long double ldexpl __P((long double, int));
344 extern long double logl __P((long double));
345 extern long double log10l __P((long double));
346 extern long double log1pl __P((long double));
347 extern long double log2l __P((long double));
348 extern long double logbl __P((long double));
349 extern long double modfl __P((long double, long double *));
350 extern long double scalbnl __P((long double, int));
351 extern long double scalblnl __P((long double, long int));

353 extern long double cbrtl __P((long double));
354 extern long double fabsl __P((long double));
355 extern long double hypotl __P((long double, long double));
356 extern long double powl __P((long double, long double));
357 extern long double sqrtl __P((long double));

359 extern long double erfl __P((long double));
360 extern long double erfcl __P((long double));
361 extern long double lgammal __P((long double));
362 extern long double tgamma __P((long double));

364 extern long double ceil __P((long double));
365 extern long double floor __P((long double));
366 extern long double nearbyint __P((long double));
367 extern long double rint __P((long double));
368 extern long int lrint __P((long double));
369 extern long double round __P((long double));
370 extern long int lround __P((long double));
371 extern long double trunc __P((long double));

373 extern long double fmod __P((long double, long double));
374 extern long double remainder __P((long double, long double));
375 extern long double remquo __P((long double, long double, int *));

377 extern long double copysignl __P((long double, long double));
378 extern long double nanl __P((const char *));
379 extern long double nextafterl __P((long double, long double));
380 extern long double nexttowardl __P((long double, long double));

382 extern long double fdiml __P((long double, long double));
383 extern long double fmaxl __P((long double, long double));
384 extern long double fminl __P((long double, long double));

383 extern long double fmal(long double, long double, long double);
386 extern long double fmal __P((long double, long double, long double));

385 #if !defined(__STRICT_STDC) && !defined(__NO_LONGLONG) || defined(__STDC_C99) || \
386     defined(__C99FEATURES__)
387 extern long long int llrint(double);
388 extern long long int llround(double);
390 extern long long int llrint __P((double));
391 extern long long int llround __P((double));

390 extern long long int llrintf(float);
391 extern long long int llroundf(float);
393 extern long long int llrintf __P((float));

```

```

394 extern long long int llroundf __P((float));

393 extern long long int llrintl(long double);
394 extern long long int llroundl(long double);
396 extern long long int llrintl __P((long double));
397 extern long long int llroundl __P((long double));
395 #endif

397 #if !defined(__cplusplus)
398 #pragma does_not_read_global_data(asinh, exp2, expml)
399 #pragma does_not_read_global_data(ilogb, log2)
400 #pragma does_not_read_global_data(scalbn, scalbln, cbrt)
401 #pragma does_not_read_global_data(erf, erfc, tgamma)
402 #pragma does_not_read_global_data(nearbyint, rint, lrint, round, lround, trunc)
403 #pragma does_not_read_global_data(remquo)
404 #pragma does_not_read_global_data(copysign, nan, nexttoward)
405 #pragma does_not_read_global_data(fdim, fmax, fmin, fma)
406 #pragma does_not_write_global_data(asinh, exp2, expml)
407 #pragma does_not_write_global_data(ilogb, log2)
408 #pragma does_not_write_global_data(scalbn, scalbln, cbrt)
409 #pragma does_not_write_global_data(erf, erfc, tgamma)
410 #pragma does_not_write_global_data(nearbyint, rint, lrint, round, lround, trunc)
411 #pragma does_not_write_global_data(copysign, nan, nexttoward)
412 #pragma does_not_write_global_data(fdim, fmax, fmin, fma)

414 #pragma does_not_read_global_data(acosf, asinf, atanf, atan2f)
415 #pragma does_not_read_global_data(cosf, sinf, tanf)
416 #pragma does_not_read_global_data(acoshf, asinhf, atanhf, coshf, sinhf, tanhf)
417 #pragma does_not_read_global_data(expf, exp2f, expmf, frexpf, ilogbf, ldexpf)
418 #pragma does_not_read_global_data(logf, log10f, log1pf, log2f, logbf)
419 #pragma does_not_read_global_data(modff, scalbnf, scalblnf)
420 #pragma does_not_read_global_data(cbrtf, fabsf, hypotf, powf, sqrtf)
421 #pragma does_not_read_global_data(erff, erfcf, lgammaf, tgammaf)
422 #pragma does_not_read_global_data(ceilf, floorf, nearbyintf)
423 #pragma does_not_read_global_data(rintf, lrintf, roundf, lroundf, truncf)
424 #pragma does_not_read_global_data(fmodf, remainderf, remquof)
425 #pragma does_not_read_global_data(copysignf, nanf, nextafterf, nexttowardf)
426 #pragma does_not_read_global_data(fdimf, fmaxf, fminf, fmaf)
427 #pragma does_not_write_global_data(acosf, asinf, atanf, atan2f)
428 #pragma does_not_write_global_data(cosf, sinf, tanf)
429 #pragma does_not_write_global_data(acoshf, asinhf, atanhf, coshf, sinhf, tanhf)
430 #pragma does_not_write_global_data(expf, exp2f, expmf, ilogbf, ldexpf)
431 #pragma does_not_write_global_data(logf, log10f, log1pf, log2f, logbf)
432 #pragma does_not_write_global_data(cbrtf, fabsf, hypotf, powf, sqrtf)
433 #pragma does_not_write_global_data(erff, erfcf, tgammaf)
434 #pragma does_not_write_global_data(ceilf, floorf, nearbyintf)
435 #pragma does_not_write_global_data(rintf, lrintf, roundf, lroundf, truncf)
436 #pragma does_not_write_global_data(fmodf, remainderf)
437 #pragma does_not_write_global_data(copysignf, nanf, nextafterf, nexttowardf)
438 #pragma does_not_write_global_data(fdimf, fmaxf, fminf, fmaf)

440 #pragma does_not_read_global_data(acosl, asinl, atanl, atan2l)
441 #pragma does_not_read_global_data(cosl, sinl, tanl)
442 #pragma does_not_read_global_data(acoshl, asinhl, atanh, coshl, sinhl, tanhl)
443 #pragma does_not_read_global_data(expl, exp2l, expml, frexpl, ilogbl, ldexpl)
444 #pragma does_not_read_global_data(logl, log10l, log1pl, log2l, logbl)
445 #pragma does_not_read_global_data(modfl, scalbnl, scalblnl)
446 #pragma does_not_read_global_data(cbrtl, fabsl, hypotl, powl, sqrtl)
447 #pragma does_not_read_global_data(erfl, erfcl, lgammal, tgamma)
448 #pragma does_not_read_global_data(ceil, floor, nearbyint)
449 #pragma does_not_read_global_data(rint, lrint, round, lround, trunc)
450 #pragma does_not_read_global_data(fmod, remainder, remquo)
451 #pragma does_not_read_global_data(copysign, nan, nextafter, nexttoward)
452 #pragma does_not_read_global_data(fdim, fmax, fmin, fma)
453 #pragma does_not_write_global_data(acosl, asinl, atanl, atan2l)
454 #pragma does_not_write_global_data(cosl, sinl, tanl)

```

```

455 #pragma does_not_write_global_data(acoshl, asinhl, atanh, coshl, sinhl, tanhl)
456 #pragma does_not_write_global_data(expl, exp2l, expml1, ilogbl, ldexpl)
457 #pragma does_not_write_global_data(logl, log10l, loglpl, log2l, logbl)
458 #pragma does_not_write_global_data(cbrtl, fabs, hypotl, powl, sqrtl)
459 #pragma does_not_write_global_data(erfl, erfcl, tgamma)
460 #pragma does_not_write_global_data(ceil, floor, nearbyint)
461 #pragma does_not_write_global_data(rintl, lrintl, roundl, lroundl, trunc)
462 #pragma does_not_write_global_data(fmodl, remainderl)
463 #pragma does_not_write_global_data(copysignl, nanl, nextafterl, nexttowardl)
464 #pragma does_not_write_global_data(fdiml, fmaxl, fminl, fmal)

466 #if !defined(_STRICT_STDC) && !defined(_NO_LONGLONG) || defined(_STDC_C99) || \
467     defined(__C99FEATURES__)
468 #pragma does_not_read_global_data(llrint, llround)
469 #pragma does_not_read_global_data(llrintf, llroundf, llrintl, llroundl)
470 #pragma does_not_write_global_data(llrint, llround)
471 #pragma does_not_write_global_data(llrintf, llroundf, llrintl, llroundl)
472 #endif
473 #endif /* !defined(__cplusplus) */

475 #if defined(__MATHERR_ERRNO_DONTCARE)
476 #pragma does_not_read_global_data(acosh, atanh, hypot, lgamma, loglp, logb)
477 #pragma does_not_read_global_data(nextafter, remainder)
478 #pragma does_not_write_global_data(acosh, atanh, hypot, loglp, logb)
479 #pragma does_not_write_global_data(nextafter, remainder)

481 #pragma no_side_effect(acosh, asinh, atanh, exp2, expml)
482 #pragma no_side_effect(ilogb, loglp, log2, logb)
483 #pragma no_side_effect(scalbn, scalbln, cbrt, hypot)
484 #pragma no_side_effect(erf, erfc, tgamma)
485 #pragma no_side_effect(nearbyint, rint, lrint, round, lround, trunc)
486 #pragma no_side_effect(remainder)
487 #pragma no_side_effect(copysign, nan, nextafter, nexttoward)
488 #pragma no_side_effect(fdim, fmax, fmin, fma)

490 #pragma no_side_effect(acosf, asinf, atanf, atan2f)
491 #pragma no_side_effect(cosf, sinf, tanf, coshf, sinhf, tanhf)
492 #pragma no_side_effect(acoshf, asinhf, atanhf, coshf, sinhf, tanhf)
493 #pragma no_side_effect(expf, exp2f, expmlf, ilogbf, ldexpf)
494 #pragma no_side_effect(logf, log10f, loglpf, log2f, logbf)
495 #pragma no_side_effect(cbrtf, fabsf, hypotf, powf, sqrtf)
496 #pragma no_side_effect(erff, erfcl, tgammaf)
497 #pragma no_side_effect(ceilf, floorf, nearbyintf)
498 #pragma no_side_effect(rintf, lrintf, roundf, lroundf, truncf)
499 #pragma no_side_effect(fmodf, remainderf)
500 #pragma no_side_effect(copysignf, nanf, nextafterf, nexttowardf)
501 #pragma no_side_effect(fdimf, fmaxf, fminf, fmaf)

503 #pragma no_side_effect(acosl, asinl, atanl, atan2l)
504 #pragma no_side_effect(cosl, sinl, tanl, coshl, sinhl, tanhl)
505 #pragma no_side_effect(acoshl, asinhl, atanh, coshl, sinhl, tanhl)
506 #pragma no_side_effect(expl, exp2l, expml1, ilogbl, ldexpl)
507 #pragma no_side_effect(logl, log10l, loglpl, log2l, logbl)
508 #pragma no_side_effect(cbrtl, fabsl, hypotl, powl, sqrtl)
509 #pragma no_side_effect(erfl, erfcl, tgamma)
510 #pragma no_side_effect(ceil, floor, nearbyint)
511 #pragma no_side_effect(rintl, lrintl, roundl, lroundl, trunc)
512 #pragma no_side_effect(fmodl, remainderl)
513 #pragma no_side_effect(copysignl, nanl, nextafterl, nexttowardl)
514 #pragma no_side_effect(fdiml, fmaxl, fminl, fmal)

516 #if !defined(_STRICT_STDC) && !defined(_NO_LONGLONG) || defined(_STDC_C99) || \
517     defined(__C99FEATURES__)
518 #pragma no_side_effect(llrint, llround, llrintf, llroundf, llrintl, llroundl)
519 #endif
520 #endif /* defined(__MATHERR_ERRNO_DONTCARE) */

```

```

521 #endif /* defined(__EXTENSIONS__) || defined(_STDC_C99) || ... */
523 #ifdef __cplusplus
524 }
    unchanged portion omitted

```

new/usr/src/head/iso/math_iso.h

1

```
*****
8375 Tue Apr 30 23:28:35 2019
new/usr/src/head/iso/math_iso.h
10882 math headers should stop supporting K&R C
*****
1 /*
2  * CDDL HEADER START
3  *
4  * The contents of this file are subject to the terms of the
5  * Common Development and Distribution License (the "License").
6  * You may not use this file except in compliance with the License.
7  *
8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9  * or http://www.opensolaris.org/os/licensing.
10 * See the License for the specific language governing permissions
11 * and limitations under the License.
12 *
13 * When distributing Covered Code, include this CDDL HEADER in each
14 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 * If applicable, add the following below this CDDL HEADER, with the
16 * fields enclosed by brackets "[]" replaced with your own identifying
17 * information: Portions Copyright [yyyy] [name of copyright owner]
18 *
19 * CDDL HEADER END
20 */
21 /*
22 * Copyright 2011 Nexenta Systems, Inc. All rights reserved.
23 */
24 /*
25 * Copyright 2004 Sun Microsystems, Inc. All rights reserved.
26 * Use is subject to license terms.
27 */

29 #ifndef _ISO_MATH_ISO_H
30 #define _ISO_MATH_ISO_H

32 #include <sys/feature_tests.h>

34 #ifdef __cplusplus
35 extern "C" {
36 #endif

38 #ifndef __P
39 #ifdef __STDC__
40 #define __P(p) p
41 #else
42 #define __P(p) ()
43 #endif
44 #endif /* !defined(__P) */

38 #if !defined(__STDC_C99) && _XOPEN_SOURCE - 0 < 600 && !defined(__C99FEATURES__)
39 typedef union _h_val {
40     unsigned long _i[sizeof (double) / sizeof (unsigned long)];
41     double _d;
42 } _h_val;

44 #ifdef __STDC__
45 extern const _h_val __huge_val;
46 #else
47 extern _h_val __huge_val;
48 #endif
49 #undef HUGE_VAL
50 #define HUGE_VAL __huge_val._d
51 #endif /* !defined(__STDC_C99) && _XOPEN_SOURCE - 0 < 600 && ... */

53 #if __cplusplus >= 199711L
```

new/usr/src/head/iso/math_iso.h

2

```
54 namespace std {
55 #endif

57 extern double acos(double);
58 extern double asin(double);
59 extern double atan(double);
60 extern double atan2(double, double);
61 extern double cos(double);
62 extern double sin(double);
63 extern double tan(double);

65 extern double cosh(double);
66 extern double sinh(double);
67 extern double tanh(double);

69 extern double exp(double);
70 extern double frexp(double, int *);
71 extern double ldexp(double, int);
72 extern double log(double);
73 extern double log10(double);
74 extern double modf(double, double *);

76 extern double pow(double, double);
77 extern double sqrt(double);

79 extern double ceil(double);
80 extern double fabs(double);
81 extern double floor(double);
82 extern double fmod(double, double);
65 extern double acos __P((double));
66 extern double asin __P((double));
67 extern double atan __P((double));
68 extern double atan2 __P((double, double));
69 extern double cos __P((double));
70 extern double sin __P((double));
71 extern double tan __P((double));

73 extern double cosh __P((double));
74 extern double sinh __P((double));
75 extern double tanh __P((double));

77 extern double exp __P((double));
78 extern double frexp __P((double, int *));
79 extern double ldexp __P((double, int));
80 extern double log __P((double));
81 extern double log10 __P((double));
82 extern double modf __P((double, double *));

84 extern double pow __P((double, double));
85 extern double sqrt __P((double));

87 extern double ceil __P((double));
88 extern double fabs __P((double));
89 extern double floor __P((double));
90 extern double fmod __P((double, double));

84 #if defined(__MATHERR_ERRNO_DONTCARE)
85 #pragma does_not_read_global_data(acos, asin, atan, atan2)
86 #pragma does_not_read_global_data(cos, sin, tan, cosh, sinh, tanh)
87 #pragma does_not_read_global_data(exp, log, log10, pow, sqrt)
88 #pragma does_not_read_global_data(frexp, ldexp, modf)
89 #pragma does_not_read_global_data(ceil, fabs, floor, fmod)
90 #pragma does_not_write_global_data(acos, asin, atan, atan2)
91 #pragma does_not_write_global_data(cos, sin, tan, cosh, sinh, tanh)
92 #pragma does_not_write_global_data(exp, log, log10, pow, sqrt)
93 #pragma does_not_write_global_data(ldexp)
```

```

94 #pragma does_not_write_global_data(ceil, fabs, floor, fmod)
95 #pragma no_side_effect(acos, asin, atan, atan2)
96 #pragma no_side_effect(cos, sin, tan, cosh, sinh, tanh)
97 #pragma no_side_effect(exp, log, log10, pow, sqrt)
98 #pragma no_side_effect(ldexp)
99 #pragma no_side_effect(ceil, fabs, floor, fmod)
100 #endif

```

```

102 #if __cplusplus >= 199711L
103 extern float  _acosf(float);
104 extern float  _asinf(float);
105 extern float  _atanf(float);
106 extern float  _atan2f(float, float);
107 extern float  _ceilf(float);
108 extern float  _cosf(float);
109 extern float  _coshf(float);
110 extern float  _expf(float);
111 extern float  _fabsf(float);
112 extern float  _floorf(float);
113 extern float  _fmodf(float, float);
114 extern float  _frexpf(float, int *);
115 extern float  _ldexpf(float, int);
116 extern float  _logf(float);
117 extern float  _log10f(float);
118 extern float  _modff(float, float *);
119 extern float  _powf(float, float);
120 extern float  _sinf(float);
121 extern float  _sinhf(float);
122 extern float  _sqrtf(float);
123 extern float  _tanf(float);
124 extern float  _tanhf(float);

```

```

126 extern long double  _acosl(long double);
127 extern long double  _asinl(long double);
128 extern long double  _atanl(long double);
129 extern long double  _atan2l(long double, long double);
130 extern long double  _ceill(long double);
131 extern long double  _cosl(long double);
132 extern long double  _coshl(long double);
133 extern long double  _expl(long double);
134 extern long double  _fabsl(long double);
135 extern long double  _floorl(long double);
136 extern long double  _fmodl(long double, long double);
137 extern long double  _frexpl(long double, int *);
138 extern long double  _ldexpl(long double, int);
139 extern long double  _logl(long double);
140 extern long double  _log10l(long double);
141 extern long double  _modfl(long double, long double *);
142 extern long double  _powl(long double, long double);
143 extern long double  _sinl(long double);
144 extern long double  _sinhl(long double);
145 extern long double  _sqrtl(long double);
146 extern long double  _tanl(long double);
147 extern long double  _tanhl(long double);

```

```

149 extern "C++" {
150 #undef  __X
151 #undef  __Y
152 inline double abs(double __X) { return fabs(__X); }

```

```

154 inline double pow(double __X, int __Y) {
155     return (pow(__X, (double)(__Y)));
156 }
161 inline double pow(double __X, int __Y) { return
162     pow(__X, (double)(__Y)); }

```

```

158 inline float abs(float __X) { return _fabsf(__X); }
159 inline float acos(float __X) { return _acosf(__X); }
160 inline float asin(float __X) { return _asinf(__X); }
161 inline float atan(float __X) { return _atanf(__X); }
162 inline float atan2(float __X, float __Y) { return _atan2f(__X, __Y); }
163 inline float ceil(float __X) { return _ceilf(__X); }
164 inline float cos(float __X) { return _cosf(__X); }
165 inline float cosh(float __X) { return _coshf(__X); }
166 inline float exp(float __X) { return _expf(__X); }
167 inline float fabs(float __X) { return _fabsf(__X); }
168 inline float floor(float __X) { return _floorf(__X); }
169 inline float fmod(float __X, float __Y) { return _fmodf(__X, __Y); }
170 inline float frexp(float __X, int *__Y) { return _frexpf(__X, __Y); }
171 inline float ldexp(float __X, int __Y) { return _ldexpf(__X, __Y); }
172 inline float log(float __X) { return _logf(__X); }
173 inline float log10(float __X) { return _log10f(__X); }
174 inline float modf(float __X, float *__Y) { return _modff(__X, __Y); }
175 inline float pow(float __X, float __Y) { return _powf(__X, __Y); }

```

```

177 inline float pow(float __X, int __Y) {
178     return (pow((double)(__X), (double)(__Y)));
179 }

```

```

182 inline float pow(float __X, int __Y) { return
183     pow((double)(__X), (double)(__Y)); }
181 inline float sin(float __X) { return _sinf(__X); }
182 inline float sinh(float __X) { return _sinhf(__X); }
183 inline float sqrt(float __X) { return _sqrtf(__X); }
184 inline float tan(float __X) { return _tanf(__X); }
185 inline float tanh(float __X) { return _tanhf(__X); }

```

```

187 inline long double abs(long double __X) { return _fabsl(__X); }
188 inline long double acos(long double __X) { return _acosl(__X); }
189 inline long double asin(long double __X) { return _asinl(__X); }
190 inline long double atan(long double __X) { return _atanl(__X); }

```

```

192 inline long double atan2(long double __X, long double __Y) {
193     return (_atan2l(__X, __Y));
194 }

```

```

194 inline long double atan2(long double __X, long double __Y) { return
195     _atan2l(__X, __Y); }
196 inline long double ceil(long double __X) { return _ceil(__X); }
197 inline long double cos(long double __X) { return _cosl(__X); }
198 inline long double cosh(long double __X) { return _coshl(__X); }
199 inline long double exp(long double __X) { return _expl(__X); }
200 inline long double fabs(long double __X) { return _fabsl(__X); }
201 inline long double floor(long double __X) { return _floorl(__X); }

```

```

203 inline long double fmod(long double __X, long double __Y) {
204     return (_fmodl(__X, __Y));
205 }

```

```

207 inline long double frexp(long double __X, int *__Y) {
208     return (_frexpl(__X, __Y));
209 }

```

```

211 inline long double ldexp(long double __X, int __Y) {
212     return (_ldexpl(__X, __Y));
213 }

```

```

202 inline long double fmod(long double __X, long double __Y) { return
203     _fmodl(__X, __Y); }
204 inline long double frexp(long double __X, int *__Y) { return
205     _frexpl(__X, __Y); }
206 inline long double ldexp(long double __X, int __Y) { return

```

```
207     __ldexpl(__X, __Y); }
215     inline long double log(long double __X) { return __logl(__X); }
216     inline long double log10(long double __X) { return __log10l(__X); }

218     inline long double modf(long double __X, long double *__Y) {
219         return (__modfl(__X, __Y));
220     }

222     inline long double pow(long double __X, long double __Y) {
223         return (__powl(__X, __Y));
224     }

226     inline long double pow(long double __X, int __Y) {
227         return (__powl(__X, (long double) (__Y)));
228     }

210     inline long double modf(long double __X, long double *__Y) { return
211         __modfl(__X, __Y); }
212     inline long double pow(long double __X, long double __Y) { return
213         __powl(__X, __Y); }
214     inline long double pow(long double __X, int __Y) { return
215         __powl(__X, (long double) (__Y)); }
230     inline long double sin(long double __X) { return __sinl(__X); }
231     inline long double sinh(long double __X) { return __sinhl(__X); }
232     inline long double sqrt(long double __X) { return __sqrtl(__X); }
233     inline long double tan(long double __X) { return __tanl(__X); }
234     inline long double tanh(long double __X) { return __tanhl(__X); }
235 }
/* end of extern "C++" */
_unchanged_portion_omitted_
```

new/usr/src/head/math.h

1

```
*****
9976 Tue Apr 30 23:28:36 2019
new/usr/src/head/math.h
10882 math headers should stop supporting K&R C
*****
1 /*
2  * CDDL HEADER START
3  *
4  * The contents of this file are subject to the terms of the
5  * Common Development and Distribution License (the "License").
6  * You may not use this file except in compliance with the License.
7  *
8  * You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
9  * or http://www.opensolaris.org/os/licensing.
10 * See the License for the specific language governing permissions
11 * and limitations under the License.
12 *
13 * When distributing Covered Code, include this CDDL HEADER in each
14 * file and include the License file at usr/src/OPENSOLARIS.LICENSE.
15 * If applicable, add the following below this CDDL HEADER, with the
16 * fields enclosed by brackets "[]" replaced with your own identifying
17 * information: Portions Copyright [yyyy] [name of copyright owner]
18 *
19 * CDDL HEADER END
20 */
21 /*
22 * Copyright 2011 Nexenta Systems, Inc. All rights reserved.
23 */
24 /*
25 * Copyright 2004 Sun Microsystems, Inc. All rights reserved.
26 * Use is subject to license terms.
27 */

29 #ifndef _MATH_H
30 #define _MATH_H

32 #include <iso/math_iso.h>
33 #include <iso/math_c99.h>

35 #if __cplusplus >= 199711L
36 using std::abs;
37 using std::acos;
38 using std::asin;
39 using std::atan2;
40 using std::atan;
41 using std::ceil;
42 using std::cos;
43 using std::cosh;
44 using std::exp;
45 using std::fabs;
46 using std::floor;
47 using std::fmod;
48 using std::frexp;
49 using std::ldexp;
50 using std::log10;
51 using std::log;
52 using std::modf;
53 using std::pow;
54 using std::sin;
55 using std::sinh;
56 using std::sqrt;
57 using std::tan;
58 using std::tanh;
59 #endif

61 #ifndef __cplusplus
```

new/usr/src/head/math.h

2

```
62 extern "C" {
63 #endif

65 #if defined(__cplusplus)
66 #define exception      __math_exception
67 #endif

69 #ifndef __P
70 #ifdef __STDC__
71 #define __P(p) p
72 #else
73 #define __P(p) ()
74 #endif
75 #endif /* !defined(__P) */

69 #if defined(__EXTENSIONS__) || defined(_XOPEN_SOURCE) || \
70     !defined(_STRICT_STDC) && !defined(_POSIX_C_SOURCE)
71 /*
72  * SVID & X/Open
73  */
74 #define M_E                2.7182818284590452354
75 #define M_LOG2E            1.44269504088889634074
76 #define M_LOG10E           0.43429448190325182765
77 #define M_LN2              0.69314718055994530942
78 #define M_LN10             2.30258509299404568402
79 #define M_PI               3.14159265358979323846
80 #define M_PI_2             1.57079632679489661923
81 #define M_PI_4             0.78539816339744830962
82 #define M_1_PI             0.31830988618379067154
83 #define M_2_PI             0.63661977236758134308
84 #define M_2_SQRTPI        1.12837916709551257390
85 #define M_SQRT2            1.41421356237309504880
86 #define M_SQRT1_2         0.70710678118654752440

88 extern int siggam;

90 #define MAXFLOAT            ((float)3.40282346638528860e+38)

92 #if defined(__EXTENSIONS__) || !defined(_XOPEN_SOURCE)
93 /*
94  * SVID
95  */
96 enum version {libm_ieee = -1, c_issue_4, ansi_1, strict_ansi};

98 #ifdef __STDC__
99 extern const enum version _lib_version;
100 #else
101 extern enum version _lib_version;
102 #endif

104 struct exception {
105     int type;
106     char *name;
107     double arg1;
108     double arg2;
109     double retval;
110 };

112 #define HUGE                MAXFLOAT

114 #define _ABS(x)              ((x) < 0 ? -(x) : (x))

116 #define _REDUCE(TYPE, X, XN, C1, C2) { \
117     double x1 = (double)(TYPE)X, x2 = X - x1; \
118     X = x1 - (XN) * (C1); X += x2; X -= (XN) * (C2); }

```

new/usr/src/head/math.h

```

120 #define DOMAIN      1
121 #define SING        2
122 #define OVERFLOW    3
123 #define UNDERFLOW  4
124 #define TLOSS       5
125 #define PLOSS       6

127 #define _POLY1(x, c) ((c)[0] * (x) + (c)[1])
128 #define _POLY2(x, c) (_POLY1((x), (c)) * (x) + (c)[2])
129 #define _POLY3(x, c) (_POLY2((x), (c)) * (x) + (c)[3])
130 #define _POLY4(x, c) (_POLY3((x), (c)) * (x) + (c)[4])
131 #define _POLY5(x, c) (_POLY4((x), (c)) * (x) + (c)[5])
132 #define _POLY6(x, c) (_POLY5((x), (c)) * (x) + (c)[6])
133 #define _POLY7(x, c) (_POLY6((x), (c)) * (x) + (c)[7])
134 #define _POLY8(x, c) (_POLY7((x), (c)) * (x) + (c)[8])
135 #define _POLY9(x, c) (_POLY8((x), (c)) * (x) + (c)[9])
136 #endif /* defined(__EXTENSIONS__) || !defined(_XOPEN_SOURCE) */

138 /*
139  * SVID & X/Open
140  */
141 /* BEGIN adopted by C99 */
142 extern double erf(double);
143 extern double erfc(double);
144 extern double hypot(double, double);
145 extern double lgamma(double);
146 extern double erf __P((double));
147 extern double erfc __P((double));
148 extern double hypot __P((double, double));
149 extern double lgamma __P((double));

151 #if defined(__MATHERR_ERRNO_DONTCARE)
152 #pragma does_not_read_global_data(erf, erfc, hypot)
153 #pragma does_not_write_global_data(erf, erfc, hypot)
154 #pragma no_side_effect(erf, erfc, hypot)
155 #endif

157 #if !defined(_STDC_C99) && _XOPEN_SOURCE - 0 < 600 && !defined(__C99FEATURES__)
158 extern int isnan(double);
159 extern int isnan __P((double));

161 #pragma does_not_read_global_data(isnan)
162 #pragma does_not_write_global_data(isnan)
163 #pragma no_side_effect(isnan)
164 #endif

166 #if defined(__EXTENSIONS__) || _XOPEN_SOURCE - 0 < 600
167 extern double gamma(double); /* deprecated; use lgamma */
168 extern double gamma __P((double)); /* deprecated; use lgamma */
169 #endif
170 extern double j0(double);
171 extern double j1(double);
172 extern double jn(int, double);
173 extern double y0(double);
174 extern double y1(double);
175 extern double yn(int, double);
176 extern double j0 __P((double));
177 extern double j1 __P((double));
178 extern double jn __P((int, double));
179 extern double y0 __P((double));
180 extern double y1 __P((double));
181 extern double yn __P((int, double));

183 #if defined(__MATHERR_ERRNO_DONTCARE)
184 #pragma does_not_read_global_data(j0, j1, jn, y0, y1, yn)

```

3

new/usr/src/head/math.h

```

174 #pragma does_not_write_global_data(j0, j1, jn, y0, y1, yn)
175 #pragma no_side_effect(j0, j1, jn, y0, y1, yn)
176 #endif
177 #if defined(__EXTENSIONS__) || !defined(_XOPEN_SOURCE) || \
178     _XOPEN_SOURCE - 0 >= 500 || \
179     defined(_XOPEN_SOURCE) && _XOPEN_SOURCE_EXTENDED - 0 == 1
180 /*
181  * SVID & XPG 4.2/5
182  */
183 extern double scalb(double, double);
184 extern double scalb __P((double, double));

186 #if defined(__MATHERR_ERRNO_DONTCARE)
187 #pragma does_not_read_global_data(scalb)
188 #pragma does_not_write_global_data(scalb)
189 #pragma no_side_effect(scalb)
190 #endif

191 /* BEGIN adopted by C99 */
192 extern double acosh(double);
193 extern double asinh(double);
194 extern double atanh(double);
195 extern double cbrt(double);
196 extern double logb(double);
197 extern double nextafter(double, double);
198 extern double remainder(double, double);
199 extern double acosh __P((double));
200 extern double asinh __P((double));
201 extern double atanh __P((double));
202 extern double cbrt __P((double));
203 extern double logb __P((double));
204 extern double nextafter __P((double, double));
205 extern double remainder __P((double, double));

207 /*
208  * XPG 4.2/5
209  */
210 extern double expml(double);
211 extern int ilogb(double);
212 extern double loglp(double);
213 extern double rint(double);
214 extern double expml __P((double));
215 extern int ilogb __P((double));
216 extern double loglp __P((double));
217 extern double rint __P((double));

219 #if defined(__MATHERR_ERRNO_DONTCARE)
220 #pragma does_not_read_global_data(acosh, asinh, atanh, cbrt)
221 #pragma does_not_read_global_data(logb, nextafter, remainder)
222 #pragma does_not_read_global_data(expml, ilogb, loglp, rint)
223 #pragma does_not_write_global_data(acosh, asinh, atanh, cbrt)
224 #pragma does_not_write_global_data(logb, nextafter, remainder)
225 #pragma no_side_effect(acosh, asinh, atanh, cbrt)
226 #pragma no_side_effect(logb, nextafter, remainder)
227 #pragma no_side_effect(expml, ilogb, loglp, rint)
228 #endif
229 /* END adopted by C99 */
230 #endif /* defined(__EXTENSIONS__) || !defined(_XOPEN_SOURCE) || ... */

232 #if defined(__EXTENSIONS__) || !defined(_XOPEN_SOURCE)
233 /*
234  * SVID
235  */
236 extern int matherr(struct exception *);
237 extern int matherr __P((struct exception *));

```

4

```

228 /*
229  * IEEE Test Vector
230 */
231 extern double significand(double);
232 extern double significand __P((double));

233 #if defined(__MATHERR_ERRNO_DONTCARE)
234 #pragma does_not_read_global_data(significand)
235 #pragma does_not_write_global_data(significand)
236 #pragma no_side_effect(significand)
237 #endif

238 extern int signgamf; /* deprecated; use signgam */
239 extern int signgaml; /* deprecated; use signgam */

240 extern int isnanf(float);
241 extern int isnanl(long double);
242 extern float gammaf(float); /* deprecated; use lgammaf */
243 extern float gammaf_r(float, int *); /* deprecated; use lgammaf_r */
244 extern float j0f(float);
245 extern float j1f(float);
246 extern float jnf(int, float);
247 extern float lgammaf_r(float, int *);
248 extern float scalbf(float, float);
249 extern float significandf(float);
250 extern float y0f(float);
251 extern float y1f(float);
252 extern float ynf(int, float);
253 extern long double gammal(long double); /* deprecated; use lgammal */
254 extern long double gammal_r(long double, int *); /* deprecated */
255 extern long double j0l(long double);
256 extern long double j1l(long double);
257 extern long double jnl(int, long double);
258 extern long double lgammal_r(long double, int *);
259 extern long double scalbl(long double, long double);
260 extern long double significandl(long double);
261 extern long double y0l(long double);
262 extern long double y1l(long double);
263 extern long double ynl(int, long double);
264 extern int isnanf __P((float));
265 extern int isnanl __P((long double));
266 extern float gammaf __P((float)); /* deprecated; use lgammaf */
267 extern float gammaf_r __P((float, int *)); /* deprecated; use lgammaf_r */
268 extern float j0f __P((float));
269 extern float j1f __P((float));
270 extern float jnf __P((int, float));
271 extern float lgammaf_r __P((float, int *));
272 extern float scalbf __P((float, float));
273 extern float significandf __P((float));
274 extern float y0f __P((float));
275 extern float y1f __P((float));
276 extern float ynf __P((int, float));
277 extern long double gammal __P((long double)); /* deprecated; use lgammal */
278 extern long double gammal_r __P((long double, int *)); /* deprecated */
279 extern long double j0l __P((long double));
280 extern long double j1l __P((long double));
281 extern long double jnl __P((int, long double));
282 extern long double lgammal_r __P((long double, int *));
283 extern long double scalbl __P((long double, long double));
284 extern long double significandl __P((long double));
285 extern long double y0l __P((long double));
286 extern long double y1l __P((long double));
287 extern long double ynl __P((int, long double));

288 #if defined(__MATHERR_ERRNO_DONTCARE)
289 #pragma does_not_read_global_data(isnanf, isnanl)
290 #pragma does_not_write_global_data(isnanf, isnanl)
291 #pragma no_side_effect(isnanf, isnanl)
292 #pragma does_not_read_global_data(gammaf_r, j0f, j1f, jnf, lgammaf_r, scalbf)
293 #pragma does_not_read_global_data(significandf, y0f, y1f, ynf)
294 #pragma does_not_write_global_data(j0f, j1f, jnf, scalbf)
295 #pragma does_not_write_global_data(significandf, y0f, y1f, ynf)
296 #pragma no_side_effect(j0f, j1f, jnf, scalbf)
297 #pragma no_side_effect(significandf, y0f, y1f, ynf)
298 #pragma does_not_read_global_data(gammal_r, j0l, j1l, jnl, lgammal_r, scalbl)
299 #pragma does_not_read_global_data(significandl, y0l, y1l, ynl)
300 #pragma does_not_write_global_data(j0l, j1l, jnl, scalbl)
301 #pragma does_not_write_global_data(significandl, y0l, y1l, ynl)
302 #pragma no_side_effect(j0l, j1l, jnl, scalbl)
303 #pragma no_side_effect(significandl, y0l, y1l, ynl)
304 #endif

305 #if defined(__MATHERR_ERRNO_DONTCARE)
306 #pragma does_not_read_global_data(sincos, sincosf, sincosl)
307 #endif

308 /* BEGIN adopted by C99 */
309 /*
310 * Functions callable from C, intended to support IEEE arithmetic.
311 */
312 extern double copysign(double, double);
313 extern double scalbn(double, int);
314 extern double copysign __P((double, double));
315 extern double scalbn __P((double, int));

316 #if defined(__MATHERR_ERRNO_DONTCARE)
317 #pragma does_not_read_global_data(copysign, scalbn)
318 #pragma does_not_write_global_data(copysign, scalbn)
319 #pragma no_side_effect(copysign, scalbn)
320 #endif

321 /* END adopted by C99 */

322 /*
323 * Reentrant version of gamma & lgamma; passes signgam back by reference
324 * as the second argument; user must allocate space for signgam.
325 */
326 extern double gamma_r(double, int *); /* deprecated; use lgamma_r */
327 extern double lgamma_r(double, int *);
328 extern double gamma_r __P((double, int *)); /* deprecated; use lgamma_r */
329 extern double lgamma_r __P((double, int *));

330 #if defined(__MATHERR_ERRNO_DONTCARE)
331 #pragma does_not_read_global_data(gamma_r, lgamma_r)
332 #endif

333 /* BEGIN adopted by C99 */
334 extern float modff(float, float *);
335 extern float modff __P((float, float *));

336 #if defined(__MATHERR_ERRNO_DONTCARE)
337 #pragma does_not_read_global_data(modff)
338 #endif

```

```

268 #pragma does_not_read_global_data(isnanf, isnanl)
269 #pragma does_not_write_global_data(isnanf, isnanl)
270 #pragma no_side_effect(isnanf, isnanl)
271 #pragma does_not_read_global_data(gammaf_r, j0f, j1f, jnf, lgammaf_r, scalbf)
272 #pragma does_not_read_global_data(significandf, y0f, y1f, ynf)
273 #pragma does_not_write_global_data(j0f, j1f, jnf, scalbf)
274 #pragma does_not_write_global_data(significandf, y0f, y1f, ynf)
275 #pragma no_side_effect(j0f, j1f, jnf, scalbf)
276 #pragma no_side_effect(significandf, y0f, y1f, ynf)
277 #pragma does_not_read_global_data(gammal_r, j0l, j1l, jnl, lgammal_r, scalbl)
278 #pragma does_not_read_global_data(significandl, y0l, y1l, ynl)
279 #pragma does_not_write_global_data(j0l, j1l, jnl, scalbl)
280 #pragma does_not_write_global_data(significandl, y0l, y1l, ynl)
281 #pragma no_side_effect(j0l, j1l, jnl, scalbl)
282 #pragma no_side_effect(significandl, y0l, y1l, ynl)
283 #endif

284 /*
285 * for sin+cos->sincos transformation
286 */
287 extern void sincos(double, double *, double *);
288 extern void sincosf(float, float *, float *);
289 extern void sincosl(long double, long double *, long double *);
290 extern void sincos __P((double, double *, double *));
291 extern void sincosf __P((float, float *, float *));
292 extern void sincosl __P((long double, long double *, long double *));

293 #if defined(__MATHERR_ERRNO_DONTCARE)
294 #pragma does_not_read_global_data(sincos, sincosf, sincosl)
295 #endif

296 /* BEGIN adopted by C99 */
297 /*
298 * Functions callable from C, intended to support IEEE arithmetic.
299 */
300 extern double copysign(double, double);
301 extern double scalbn(double, int);
302 extern double copysign __P((double, double));
303 extern double scalbn __P((double, int));

304 #if defined(__MATHERR_ERRNO_DONTCARE)
305 #pragma does_not_read_global_data(copysign, scalbn)
306 #pragma does_not_write_global_data(copysign, scalbn)
307 #pragma no_side_effect(copysign, scalbn)
308 #endif

309 /* END adopted by C99 */

310 /*
311 * Reentrant version of gamma & lgamma; passes signgam back by reference
312 * as the second argument; user must allocate space for signgam.
313 */
314 extern double gamma_r(double, int *); /* deprecated; use lgamma_r */
315 extern double lgamma_r(double, int *);
316 extern double gamma_r __P((double, int *)); /* deprecated; use lgamma_r */
317 extern double lgamma_r __P((double, int *));

318 #if defined(__MATHERR_ERRNO_DONTCARE)
319 #pragma does_not_read_global_data(gamma_r, lgamma_r)
320 #endif

321 /* BEGIN adopted by C99 */
322 extern float modff(float, float *);
323 extern float modff __P((float, float *));

324 #if defined(__MATHERR_ERRNO_DONTCARE)
325 #pragma does_not_read_global_data(modff)
326 #endif

```


new/usr/src/head/math.h

7

```
326 #endif
327 /* END adopted by C99 */

329 #if defined(__EXTENSIONS__) || !defined(__cplusplus)
330 #include <floatingpoint.h>
331 #endif
332 #endif /* defined(__EXTENSIONS__) || !defined(_XOPEN_SOURCE) */
333 #endif /* defined(__EXTENSIONS__) || defined(_XOPEN_SOURCE) || ... */

335 #if defined(__cplusplus) && defined(__GNUC__)
336 #undef exception
337 #endif

339 #ifdef __cplusplus
340 }
_____unchanged_portion_omitted_
```