#1 "../../../../../../home/drsatya/Desktop/lab1/mat\_pipeline.cpp"

#1 "../../../../../../home/drsatya/Desktop/lab1/mat\_pipeline.cpp" 1

#1 "<built-in>" 1

#1 "<built-in>" 3

#155 "<built-in>" 3

#1 "<command line>" 1

#1 "/opt/Xilinx/Vivado\_HLS/2017.2/common/technology/autopilot/etc/autopilot\_ssdm\_op.h" 1

/\* autopilot\_ssdm\_op.h\*/

/\*

#- (c) Copyright 2011-2017 Xilinx, Inc. All rights reserved.

#-

#- This file contains confidential and proprietary information

#- of Xilinx, Inc. and is protected under U.S. and

#- international copyright and other intellectual property

#- laws.

#-

#- DISCLAIMER

#- This disclaimer is not a license and does not grant any

#- rights to the materials distributed herewith. Except as

#- otherwise provided in a valid license issued to you by

#- Xilinx, and to the maximum extent permitted by applicable

#- law: (1) THESE MATERIALS ARE MADE AVAILABLE "AS IS" AND

#- WITH ALL FAULTS, AND XILINX HEREBY DISCLAIMS ALL WARRANTIES

#- AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING

#- BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-

#- INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and

#- (2) Xilinx shall not be liable (whether in contract or tort,

#- including negligence, or under any other theory of

#- liability) for any loss or damage of any kind or nature

#- related to, arising under or in connection with these

#- materials, including for any direct, or any indirect,

#- special, incidental, or consequential loss or damage

#- (including loss of data, profits, goodwill, or any type of

#- loss or damage suffered as a result of any action brought

#- by a third party) even if such damage or loss was

#- reasonably foreseeable or Xilinx had been advised of the

#- possibility of the same.

#-

#- CRITICAL APPLICATIONS

#- Xilinx products are not designed or intended to be fail-

#- safe, or for use in any application requiring fail-safe

#- performance, such as life-support or safety devices or

#- systems, Class III medical devices, nuclear facilities,

#- applications related to the deployment of airbags, or any

#- other applications that could lead to death, personal

#- injury, or severe property or environmental damage

#- (individually and collectively, "Critical

#- Applications"). Customer assumes the sole risk and

#- liability of any use of Xilinx products in Critical

#- Applications, subject only to applicable laws and

#- regulations governing limitations on product liability.

#-

#- THIS COPYRIGHT NOTICE AND DISCLAIMER MUST BE RETAINED AS

#- PART OF THIS FILE AT ALL TIMES.

#- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* $Id$

\*/

#145 "/opt/Xilinx/Vivado\_HLS/2017.2/common/technology/autopilot/etc/autopilot\_ssdm\_op.h"

/\*#define AP\_SPEC\_ATTR \_\_attribute\_\_ ((pure))\*/

//adu: patched

#156 "/opt/Xilinx/Vivado\_HLS/2017.2/common/technology/autopilot/etc/autopilot\_ssdm\_op.h"

extern "C" {

/\*\*\*\*\*\* SSDM Intrinsics: OPERATIONS \*\*\*/

// Interface operations

//typedef unsigned int \_\_attribute\_\_ ((bitwidth(1))) \_uint1\_;

typedef bool \_uint1\_;

void \_ssdm\_op\_IfRead(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_IfWrite(...) \_\_attribute\_\_ ((nothrow));

\_uint1\_ \_ssdm\_op\_IfNbRead(...) \_\_attribute\_\_ ((nothrow));

\_uint1\_ \_ssdm\_op\_IfNbWrite(...) \_\_attribute\_\_ ((nothrow));

\_uint1\_ \_ssdm\_op\_IfCanRead(...) \_\_attribute\_\_ ((nothrow));

\_uint1\_ \_ssdm\_op\_IfCanWrite(...) \_\_attribute\_\_ ((nothrow));

// Stream Intrinsics

void \_ssdm\_StreamRead(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_StreamWrite(...) \_\_attribute\_\_ ((nothrow));

\_uint1\_ \_ssdm\_StreamNbRead(...) \_\_attribute\_\_ ((nothrow));

\_uint1\_ \_ssdm\_StreamNbWrite(...) \_\_attribute\_\_ ((nothrow));

\_uint1\_ \_ssdm\_StreamCanRead(...) \_\_attribute\_\_ ((nothrow));

\_uint1\_ \_ssdm\_StreamCanWrite(...) \_\_attribute\_\_ ((nothrow));

unsigned \_ssdm\_StreamSize(...) \_\_attribute\_\_ ((nothrow));

// Misc

void \_ssdm\_op\_MemShiftRead(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_Wait(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_Poll(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_Return(...) \_\_attribute\_\_ ((nothrow));

/\* SSDM Intrinsics: SPECIFICATIONS \*/

void \_ssdm\_op\_SpecSynModule(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecTopModule(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecProcessDecl(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecProcessDef(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecPort(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecConnection(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecChannel(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecSensitive(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecModuleInst(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecPortMap(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecReset(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecPlatform(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecClockDomain(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecPowerDomain(...) \_\_attribute\_\_ ((nothrow));

int \_ssdm\_op\_SpecRegionBegin(...) \_\_attribute\_\_ ((nothrow));

int \_ssdm\_op\_SpecRegionEnd(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecLoopName(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecLoopTripCount(...) \_\_attribute\_\_ ((nothrow));

int \_ssdm\_op\_SpecStateBegin(...) \_\_attribute\_\_ ((nothrow));

int \_ssdm\_op\_SpecStateEnd(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecInterface(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecPipeline(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecDataflowPipeline(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecLatency(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecParallel(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecProtocol(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecOccurrence(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecResource(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecResourceLimit(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecCHCore(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecFUCore(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecIFCore(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecIPCore(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecKeepValue(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecMemCore(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecExt(...) \_\_attribute\_\_ ((nothrow));

/\*void\* \_ssdm\_op\_SpecProcess(...) SSDM\_SPEC\_ATTR;

void\* \_ssdm\_op\_SpecEdge(...) SSDM\_SPEC\_ATTR; \*/

/\* Presynthesis directive functions \*/

void \_ssdm\_SpecArrayDimSize(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_RegionBegin(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_RegionEnd(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_Unroll(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_UnrollRegion(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_InlineAll(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_InlineLoop(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_Inline(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_InlineSelf(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_InlineRegion(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecArrayMap(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecArrayPartition(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecArrayReshape(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecStream(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecExpr(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecExprBalance(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecDependence(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecLoopMerge(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecLoopFlatten(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecLoopRewind(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecFuncInstantiation(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecFuncBuffer(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecFuncExtract(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecConstant(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_DataPack(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_SpecDataPack(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecBitsMap(...) \_\_attribute\_\_ ((nothrow));

void \_ssdm\_op\_SpecLicense(...) \_\_attribute\_\_ ((nothrow));

void \_\_xilinx\_ip\_top(...) \_\_attribute\_\_ ((nothrow));

}

#413 "/opt/Xilinx/Vivado\_HLS/2017.2/common/technology/autopilot/etc/autopilot\_ssdm\_op.h"

/\*#define \_ssdm\_op\_WaitUntil(X) while (!(X)) \_ssdm\_op\_Wait(1);

#define \_ssdm\_op\_Delayed(X) X \*/

#427 "/opt/Xilinx/Vivado\_HLS/2017.2/common/technology/autopilot/etc/autopilot\_ssdm\_op.h"

// 67d7842dbbe25473c3c32b93c0da8047785f30d78e8a024de1b57352245f9689

#7 "<command line>" 2

#1 "<built-in>" 2

#1 "../../../../../../home/drsatya/Desktop/lab1/mat\_pipeline.cpp" 2

#1 "../../../../../../home/drsatya/Desktop/lab1/matrixmul.h" 1

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// (c) Copyright 2009 - 2012 Xilinx, Inc. All rights reserved.

//

// This file contains confidential and proprietary information

// of Xilinx, Inc. and is protected under U.S. and

// international copyright and other intellectual property

// laws.

//

// DISCLAIMER

// This disclaimer is not a license and does not grant any

// rights to the materials distributed herewith. Except as

// otherwise provided in a valid license issued to you by

// Xilinx, and to the maximum extent permitted by applicable

// law: (1) THESE MATERIALS ARE MADE AVAILABLE "AS IS" AND

// WITH ALL FAULTS, AND XILINX HEREBY DISCLAIMS ALL WARRANTIES

// AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING

// BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-

// INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and

// (2) Xilinx shall not be liable (whether in contract or tort,

// liability) for any loss or damage of any kind or nature

// related to, arising under or in connection with these

// materials, including for any direct, or any indirect,

// special, incidental, or consequential loss or damage

// (including loss of data, profits, goodwill, or any type of

// loss or damage suffered as a result of any action brought

// by a third party) even if such damage or loss was

// reasonably foreseeable or Xilinx had been advised of the

// possibility of the same.

//

// CRITICAL APPLICATIONS

// Xilinx products are not designed or intended to be fail-

// safe, or for use in any application requiring fail-safe

// performance, such as life-support or safety devices or

// systems, Class III medical devices, nuclear facilities,

// applications related to the deployment of airbags, or any

// other applications that could lead to death, personal

// injury, or severe property or environmental damage

// (individually and collectively, "Critical

// Applications"). Customer assumes the sole risk and

// liability of any use of Xilinx products in Critical

// Applications, subject only to applicable laws and

// regulations governing limitations on product liability.

//

// THIS COPYRIGHT NOTICE AND DISCLAIMER MUST BE RETAINED AS

// PART OF THIS FILE AT ALL TIMES.

//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// \_\_\_\_ \_\_\_\_

// / /\/ /

// /\_\_\_/ \ / Vendor: Xilinx

// \ \ \/ Version: %version

// \ \ Application: AutoESL

// / / Filename: matrixmul.h

// /\_\_\_/ /\ Date Last Modified: $Date: 2012/3/30 18:53:07 $

// \ \ / \ Date Created: Fri Mar 30 2012

// \\_\_\_\/\\_\_\_//

//Device: All

//Design Name: maxtrixmul1

//Purpose:

// This is the header for the matrixmul.cpp design.

//Reference:

//Revision History:

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#1 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/cmath" 1 3

// -\*- C++ -\*- C forwarding header.

// Copyright (C) 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005,

// 2006, 2007, 2008, 2009, 2010, 2011

// Free Software Foundation, Inc.

//

// This file is part of the GNU ISO C++ Library. This library is free

// software; you can redistribute it and/or modify it under the

// terms of the GNU General Public License as published by the

// Free Software Foundation; either version 3, or (at your option)

// any later version.

// This library is distributed in the hope that it will be useful,

// but WITHOUT ANY WARRANTY; without even the implied warranty of

// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

// GNU General Public License for more details.

// Under Section 7 of GPL version 3, you are granted additional

// permissions described in the GCC Runtime Library Exception, version

// 3.1, as published by the Free Software Foundation.

// You should have received a copy of the GNU General Public License and

// a copy of the GCC Runtime Library Exception along with this program;

// see the files COPYING3 and COPYING.RUNTIME respectively. If not, see

// <http://www.gnu.org/licenses/>.

/\*\* @file include/cmath

\* This is a Standard C++ Library file. You should @c \#include this file

\* in your programs, rather than any of the @a \*.h implementation files.

\*

\* This is the C++ version of the Standard C Library header @c math.h,

\* and its contents are (mostly) the same as that header, but are all

\* contained in the namespace @c std (except for names which are defined

\* as macros in C).

\*/

//

// ISO C++ 14882: 26.5 C library

//

#41 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/cmath" 3

#1 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 1 3

// Predefined symbols and macros -\*- C++ -\*-

// Copyright (C) 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005,

// 2006, 2007, 2008, 2009, 2010, 2011 Free Software Foundation, Inc.

//

// This file is part of the GNU ISO C++ Library. This library is free

// software; you can redistribute it and/or modify it under the

// terms of the GNU General Public License as published by the

// Free Software Foundation; either version 3, or (at your option)

// any later version.

// This library is distributed in the hope that it will be useful,

// but WITHOUT ANY WARRANTY; without even the implied warranty of

// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

// GNU General Public License for more details.

// Under Section 7 of GPL version 3, you are granted additional

// permissions described in the GCC Runtime Library Exception, version

// 3.1, as published by the Free Software Foundation.

// You should have received a copy of the GNU General Public License and

// a copy of the GCC Runtime Library Exception along with this program;

// see the files COPYING3 and COPYING.RUNTIME respectively. If not, see

// <http://www.gnu.org/licenses/>.

/\*\* @file bits/c++config.h

\* This is an internal header file, included by other library headers.

\* Do not attempt to use it directly. @headername{iosfwd}

\*/

// The current version of the C++ library in compressed ISO date format.

// Macros for various attributes.

// \_GLIBCXX\_PURE

// \_GLIBCXX\_CONST

// \_GLIBCXX\_NORETURN

// \_GLIBCXX\_NOTHROW

// \_GLIBCXX\_VISIBILITY

#63 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 3

// Macros for visibility attributes.

// \_GLIBCXX\_HAVE\_ATTRIBUTE\_VISIBILITY

// \_GLIBCXX\_VISIBILITY

#76 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 3

// Macros for deprecated attributes.

// \_GLIBCXX\_USE\_DEPRECATED

// \_GLIBCXX\_DEPRECATED

#91 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 3

// Macro for constexpr, to support in mixed 03/0x mode.

#102 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 3

// Macro for extern template, ie controling template linkage via use

// of extern keyword on template declaration. As documented in the g++

// manual, it inhibits all implicit instantiations and is used

// throughout the library to avoid multiple weak definitions for

// required types that are already explicitly instantiated in the

// library binary. This substantially reduces the binary size of

// resulting executables.

// Special case: \_GLIBCXX\_EXTERN\_TEMPLATE == -1 disallows extern

// templates only in basic\_string, thus activating its debug-mode

// checks even at -O0.

/\*

Outline of libstdc++ namespaces.

namespace std

{

namespace \_\_debug { }

namespace \_\_parallel { }

namespace \_\_profile { }

namespace \_\_cxx1998 { }

namespace \_\_detail { }

namespace rel\_ops { }

namespace tr1

{

namespace placeholders { }

namespace regex\_constants { }

namespace \_\_detail { }

}

namespace decimal { }

namespace chrono { }

namespace placeholders { }

namespace regex\_constants { }

namespace this\_thread { }

}

namespace abi { }

namespace \_\_gnu\_cxx

{

namespace \_\_detail { }

}

For full details see:

http://gcc.gnu.org/onlinedocs/libstdc++/latest-doxygen/namespaces.html

\*/

namespace std

{

typedef long unsigned int size\_t;

typedef long int ptrdiff\_t;

}

// Defined if inline namespaces are used for versioning.

// Inline namespace for symbol versioning.

#208 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 3

// Inline namespaces for special modes: debug, parallel, profile.

#255 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 3

// Macros for namespace scope. Either namespace std:: or the name

// of some nested namespace within it corresponding to the active mode.

// \_GLIBCXX\_STD\_A

// \_GLIBCXX\_STD\_C

//

// Macros for opening/closing conditional namespaces.

// \_GLIBCXX\_BEGIN\_NAMESPACE\_ALGO

// \_GLIBCXX\_END\_NAMESPACE\_ALGO

// \_GLIBCXX\_BEGIN\_NAMESPACE\_CONTAINER

// \_GLIBCXX\_END\_NAMESPACE\_CONTAINER

#307 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 3

// GLIBCXX\_ABI Deprecated

// Define if compatibility should be provided for -mlong-double-64.

// Inline namespace for long double 128 mode.

#326 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 3

// Assert.

#352 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 3

// Macros for race detectors.

// \_GLIBCXX\_SYNCHRONIZATION\_HAPPENS\_BEFORE(A) and

// \_GLIBCXX\_SYNCHRONIZATION\_HAPPENS\_AFTER(A) should be used to explain

// atomic (lock-free) synchronization to race detectors:

// the race detector will infer a happens-before arc from the former to the

// latter when they share the same argument pointer.

//

// The most frequent use case for these macros (and the only case in the

// current implementation of the library) is atomic reference counting:

// void \_M\_remove\_reference()

// {

// \_GLIBCXX\_SYNCHRONIZATION\_HAPPENS\_BEFORE(&this->\_M\_refcount);

// if (\_\_gnu\_cxx::\_\_exchange\_and\_add\_dispatch(&this->\_M\_refcount, -1) <= 0)

// {

// \_GLIBCXX\_SYNCHRONIZATION\_HAPPENS\_AFTER(&this->\_M\_refcount);

// \_M\_destroy(\_\_a);

// }

// }

// The annotations in this example tell the race detector that all memory

// accesses occurred when the refcount was positive do not race with

// memory accesses which occurred after the refcount became zero.

// Macros for C linkage: define extern "C" linkage only when using C++.

#390 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 3

// First includes.

// Pick up any OS-specific definitions.

#1 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/os\_defines.h" 1 3

// Specific definitions for GNU/Linux -\*- C++ -\*-

// Copyright (C) 2000, 2001, 2002, 2003, 2009, 2010

// Free Software Foundation, Inc.

//

// This file is part of the GNU ISO C++ Library. This library is free

// software; you can redistribute it and/or modify it under the

// terms of the GNU General Public License as published by the

// Free Software Foundation; either version 3, or (at your option)

// any later version.

// This library is distributed in the hope that it will be useful,

// but WITHOUT ANY WARRANTY; without even the implied warranty of

// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

// GNU General Public License for more details.

// Under Section 7 of GPL version 3, you are granted additional

// permissions described in the GCC Runtime Library Exception, version

// 3.1, as published by the Free Software Foundation.

// You should have received a copy of the GNU General Public License and

// a copy of the GCC Runtime Library Exception along with this program;

// see the files COPYING3 and COPYING.RUNTIME respectively. If not, see

// <http://www.gnu.org/licenses/>.

/\*\* @file bits/os\_defines.h

\* This is an internal header file, included by other library headers.

\* Do not attempt to use it directly. @headername{iosfwd}

\*/

// System-specific #define, typedefs, corrections, etc, go here. This

// file will come before all others.

// This keeps isanum, et al from being propagated as macros.

#1 "/usr/include/features.h" 1 3 4

/\* Copyright (C) 1991-1993,1995-2007,2009,2010,2011

Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

/\* These are defined by the user (or the compiler)

to specify the desired environment:

\_\_STRICT\_ANSI\_\_ ISO Standard C.

\_ISOC99\_SOURCE Extensions to ISO C89 from ISO C99.

\_POSIX\_SOURCE IEEE Std 1003.1.

\_POSIX\_C\_SOURCE If ==1, like \_POSIX\_SOURCE; if >=2 add IEEE Std 1003.2;

if >=199309L, add IEEE Std 1003.1b-1993;

if >=199506L, add IEEE Std 1003.1c-1995;

if >=200112L, all of IEEE 1003.1-2004

if >=200809L, all of IEEE 1003.1-2008

\_XOPEN\_SOURCE Includes POSIX and XPG things. Set to 500 if

Single Unix conformance is wanted, to 600 for the

sixth revision, to 700 for the seventh revision.

\_XOPEN\_SOURCE\_EXTENDED XPG things and X/Open Unix extensions.

\_LARGEFILE\_SOURCE Some more functions for correct standard I/O.

\_LARGEFILE64\_SOURCE Additional functionality from LFS for large files.

\_FILE\_OFFSET\_BITS=N Select default filesystem interface.

\_BSD\_SOURCE ISO C, POSIX, and 4.3BSD things.

\_SVID\_SOURCE ISO C, POSIX, and SVID things.

\_ATFILE\_SOURCE Additional \*at interfaces.

\_GNU\_SOURCE All of the above, plus GNU extensions.

\_REENTRANT Select additionally reentrant object.

\_THREAD\_SAFE Same as \_REENTRANT, often used by other systems.

\_FORTIFY\_SOURCE If set to numeric value > 0 additional security

measures are defined, according to level.

The `-ansi' switch to the GNU C compiler defines \_\_STRICT\_ANSI\_\_.

If none of these are defined, the default is to have \_SVID\_SOURCE,

\_BSD\_SOURCE, and \_POSIX\_SOURCE set to one and \_POSIX\_C\_SOURCE set to

200112L. If more than one of these are defined, they accumulate.

For example \_\_STRICT\_ANSI\_\_, \_POSIX\_SOURCE and \_POSIX\_C\_SOURCE

together give you ISO C, 1003.1, and 1003.2, but nothing else.

These are defined by this file and are used by the

header files to decide what to declare or define:

\_\_USE\_ISOC99 Define ISO C99 things.

\_\_USE\_ISOC95 Define ISO C90 AMD1 (C95) things.

\_\_USE\_POSIX Define IEEE Std 1003.1 things.

\_\_USE\_POSIX2 Define IEEE Std 1003.2 things.

\_\_USE\_POSIX199309 Define IEEE Std 1003.1, and .1b things.

\_\_USE\_POSIX199506 Define IEEE Std 1003.1, .1b, .1c and .1i things.

\_\_USE\_XOPEN Define XPG things.

\_\_USE\_XOPEN\_EXTENDED Define X/Open Unix things.

\_\_USE\_UNIX98 Define Single Unix V2 things.

\_\_USE\_XOPEN2K Define XPG6 things.

\_\_USE\_XOPEN2KXSI Define XPG6 XSI things.

\_\_USE\_XOPEN2K8 Define XPG7 things.

\_\_USE\_XOPEN2K8XSI Define XPG7 XSI things.

\_\_USE\_LARGEFILE Define correct standard I/O things.

\_\_USE\_LARGEFILE64 Define LFS things with separate names.

\_\_USE\_FILE\_OFFSET64 Define 64bit interface as default.

\_\_USE\_BSD Define 4.3BSD things.

\_\_USE\_SVID Define SVID things.

\_\_USE\_MISC Define things common to BSD and System V Unix.

\_\_USE\_ATFILE Define \*at interfaces and AT\_\* constants for them.

\_\_USE\_GNU Define GNU extensions.

\_\_USE\_REENTRANT Define reentrant/thread-safe \*\_r functions.

\_\_USE\_FORTIFY\_LEVEL Additional security measures used, according to level.

\_\_FAVOR\_BSD Favor 4.3BSD things in cases of conflict.

The macros `\_\_GNU\_LIBRARY\_\_', `\_\_GLIBC\_\_', and `\_\_GLIBC\_MINOR\_\_' are

defined by this file unconditionally. `\_\_GNU\_LIBRARY\_\_' is provided

only for compatibility. All new code should use the other symbols

to test for features.

All macros listed above as possibly being defined by this file are

explicitly undefined if they are not explicitly defined.

Feature-test macros that are not defined by the user or compiler

but are implied by the other feature-test macros defined (or by the

lack of any definitions) are defined by the file. \*/

/\* Undefine everything, so we get a clean slate. \*/

#124 "/usr/include/features.h" 3 4

/\* Suppress kernel-name space pollution unless user expressedly asks

for it. \*/

/\* Always use ISO C things. \*/

/\* Convenience macros to test the versions of glibc and gcc.

Use them like this:

#if \_\_GNUC\_PREREQ (2,8)

... code requiring gcc 2.8 or later ...

#endif

Note - they won't work for gcc1 or glibc1, since the \_MINOR macros

were not defined then. \*/

#148 "/usr/include/features.h" 3 4

/\* If \_BSD\_SOURCE was defined by the user, favor BSD over POSIX. \*/

/\* If \_GNU\_SOURCE was defined by the user, turn on all the other features. \*/

#179 "/usr/include/features.h" 3 4

/\* If nothing (other than \_GNU\_SOURCE) is defined,

define \_BSD\_SOURCE and \_SVID\_SOURCE. \*/

/\* This is to enable the ISO C99 extension. Also recognize the old macro

which was used prior to the standard acceptance. This macro will

eventually go away and the features enabled by default once the ISO C99

standard is widely adopted. \*/

/\* This is to enable the ISO C90 Amendment 1:1995 extension. \*/

/\* If none of the ANSI/POSIX macros are defined, use POSIX.1 and POSIX.2

(and IEEE Std 1003.1b-1993 unless \_XOPEN\_SOURCE is defined). \*/

#323 "/usr/include/features.h" 3 4

/\* Define \_\_STDC\_IEC\_559\_\_ and other similar macros. \*/

#1 "/usr/include/x86\_64-linux-gnu/bits/predefs.h" 1 3 4

/\* Copyright (C) 2005 Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

#26 "/usr/include/x86\_64-linux-gnu/bits/predefs.h" 3 4

/\* We do support the IEC 559 math functionality, real and complex. \*/

#325 "/usr/include/features.h" 2 3 4

/\* wchar\_t uses ISO 10646-1 (2nd ed., published 2000-09-15) / Unicode 3.1. \*/

/\* This macro indicates that the installed library is the GNU C Library.

For historic reasons the value now is 6 and this will stay from now

on. The use of this variable is deprecated. Use \_\_GLIBC\_\_ and

\_\_GLIBC\_MINOR\_\_ now (see below) when you want to test for a specific

GNU C library version and use the values in <gnu/lib-names.h> to get

the sonames of the shared libraries. \*/

/\* Major and minor version number of the GNU C library package. Use

these macros to test for features in specific releases. \*/

/\* Decide whether a compiler supports the long long datatypes. \*/

/\* This is here only because every header file already includes this one. \*/

#1 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 1 3 4

/\* Copyright (C) 1992-2001, 2002, 2004, 2005, 2006, 2007, 2009, 2011

Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

/\* We are almost always included from features.h. \*/

/\* The GNU libc does not support any K&R compilers or the traditional mode

of ISO C compilers anymore. Check for some of the combinations not

anymore supported. \*/

/\* Some user header file might have defined this before. \*/

/\* All functions, except those with callbacks or those that

synchronize memory, are leaf functions. \*/

#51 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

/\* GCC can always grok prototypes. For C++ programs we add throw()

to help it optimize the function calls. But this works only with

gcc 2.8.x and egcs. For gcc 3.2 and up we even mark C functions

as non-throwing using a function attribute since programs can use

the -fexceptions options for C code as well. \*/

#86 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

/\* These two macros are not used in glibc anymore. They are kept here

only because some other projects expect the macros to be defined. \*/

/\* For these things, GCC behaves the ANSI way normally,

and the non-ANSI way under -traditional. \*/

/\* This is not a typedef so `const \_\_ptr\_t' does the right thing. \*/

/\* C++ needs to know that types and declarations are C, not C++. \*/

#112 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

/\* The standard library needs the functions from the ISO C90 standard

in the std namespace. At the same time we want to be safe for

future changes and we include the ISO C99 code in the non-standard

namespace \_\_c99. The C++ wrapper header take case of adding the

definitions to the global namespace. \*/

#125 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

/\* For compatibility we do not add the declarations into any

namespace. They will end up in the global namespace which is what

old code expects. \*/

#137 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

/\* Support for bounded pointers. \*/

/\* Fortify support. \*/

#161 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

/\* Support for flexible arrays. \*/

/\* GCC 2.97 supports C99 flexible array members. \*/

#179 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

/\* \_\_asm\_\_ ("xyz") is used throughout the headers to rename functions

at the assembly language level. This is wrapped by the \_\_REDIRECT

macro, in order to support compilers that can do this some other

way. When compilers don't support asm-names at all, we have to do

preprocessor tricks instead (which don't have exactly the right

semantics, but it's the best we can do).

Example:

int \_\_REDIRECT(setpgrp, (\_\_pid\_t pid, \_\_pid\_t pgrp), setpgid); \*/

#206 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

/\*

#elif \_\_SOME\_OTHER\_COMPILER\_\_

# define \_\_REDIRECT(name, proto, alias) name proto; \

\_Pragma("let " #name " = " #alias)

\*/

/\* GCC has various useful declarations that can be made with the

`\_\_attribute\_\_' syntax. All of the ways we use this do fine if

they are omitted for compilers that don't understand it. \*/

/\* At some point during the gcc 2.96 development the `malloc' attribute

for functions was introduced. We don't want to use it unconditionally

(although this would be possible) since it generates warnings. \*/

/\* At some point during the gcc 2.96 development the `pure' attribute

for functions was introduced. We don't want to use it unconditionally

(although this would be possible) since it generates warnings. \*/

/\* This declaration tells the compiler that the value is constant. \*/

/\* At some point during the gcc 3.1 development the `used' attribute

for functions was introduced. We don't want to use it unconditionally

(although this would be possible) since it generates warnings. \*/

#257 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

/\* gcc allows marking deprecated functions. \*/

/\* At some point during the gcc 2.8 development the `format\_arg' attribute

for functions was introduced. We don't want to use it unconditionally

(although this would be possible) since it generates warnings.

If several `format\_arg' attributes are given for the same function, in

gcc-3.0 and older, all but the last one are ignored. In newer gccs,

all designated arguments are considered. \*/

/\* At some point during the gcc 2.97 development the `strfmon' format

attribute for functions was introduced. We don't want to use it

unconditionally (although this would be possible) since it

generates warnings. \*/

/\* The nonull function attribute allows to mark pointer parameters which

must not be NULL. \*/

/\* If fortification mode, we warn about unused results of certain

function calls which can lead to problems. \*/

#310 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

/\* Forces a function to be always inlined. \*/

/\* GCC 4.3 and above with -std=c99 or -std=gnu99 implements ISO C99

inline semantics, unless -fgnu89-inline is used. \*/

#340 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

/\* GCC 4.3 and above allow passing all anonymous arguments of an

\_\_extern\_always\_inline function to some other vararg function. \*/

/\* It is possible to compile containing GCC extensions even if GCC is

run in pedantic mode if the uses are carefully marked using the

`\_\_extension\_\_' keyword. But this is not generally available before

version 2.8. \*/

/\* \_\_restrict is known in EGCS 1.2 and above. \*/

/\* ISO C99 also allows to declare arrays as non-overlapping. The syntax is

array\_name[restrict]

GCC 3.1 supports this. \*/

#378 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 3 4

#1 "/usr/include/x86\_64-linux-gnu/bits/wordsize.h" 1 3 4

/\* Determine the wordsize from the preprocessor defines. \*/

#379 "/usr/include/x86\_64-linux-gnu/sys/cdefs.h" 2 3 4

#358 "/usr/include/features.h" 2 3 4

/\* If we don't have \_\_REDIRECT, prototypes will be missing if

\_\_USE\_FILE\_OFFSET64 but not \_\_USE\_LARGEFILE[64]. \*/

/\* Decide whether we can define 'extern inline' functions in headers. \*/

/\* There are some functions that must be declared 'extern inline' even with

-Os when building LIBC, or they'll end up undefined. \*/

/\* This is here only because every header file already includes this one.

Get the definitions of all the appropriate `\_\_stub\_FUNCTION' symbols.

<gnu/stubs.h> contains `#define \_\_stub\_FUNCTION' when FUNCTION is a stub

that will always return failure (and set errno to ENOSYS). \*/

#1 "/usr/include/x86\_64-linux-gnu/gnu/stubs.h" 1 3 4

/\* This file selects the right generated file of `\_\_stub\_FUNCTION' macros

based on the architecture being compiled for. \*/

#1 "/usr/include/x86\_64-linux-gnu/bits/wordsize.h" 1 3 4

/\* Determine the wordsize from the preprocessor defines. \*/

#5 "/usr/include/x86\_64-linux-gnu/gnu/stubs.h" 2 3 4

#1 "/usr/include/x86\_64-linux-gnu/gnu/stubs-64.h" 1 3 4

/\* This file is automatically generated.

It defines a symbol `\_\_stub\_FUNCTION' for each function

in the C library which is a stub, meaning it will fail

every time called, usually setting errno to ENOSYS. \*/

#10 "/usr/include/x86\_64-linux-gnu/gnu/stubs.h" 2 3 4

#390 "/usr/include/features.h" 2 3 4

#41 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/os\_defines.h" 2 3

#394 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 2 3

// Pick up any CPU-specific definitions.

#1 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/cpu\_defines.h" 1 3

// Specific definitions for generic platforms -\*- C++ -\*-

// Copyright (C) 2005, 2009, 2010 Free Software Foundation, Inc.

//

// This file is part of the GNU ISO C++ Library. This library is free

// software; you can redistribute it and/or modify it under the

// terms of the GNU General Public License as published by the

// Free Software Foundation; either version 3, or (at your option)

// any later version.

// This library is distributed in the hope that it will be useful,

// but WITHOUT ANY WARRANTY; without even the implied warranty of

// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

// GNU General Public License for more details.

// Under Section 7 of GPL version 3, you are granted additional

// permissions described in the GCC Runtime Library Exception, version

// 3.1, as published by the Free Software Foundation.

// You should have received a copy of the GNU General Public License and

// a copy of the GCC Runtime Library Exception along with this program;

// see the files COPYING3 and COPYING.RUNTIME respectively. If not, see

// <http://www.gnu.org/licenses/>.

/\*\* @file bits/cpu\_defines.h

\* This is an internal header file, included by other library headers.

\* Do not attempt to use it directly. @headername{iosfwd}

\*/

#397 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/x86\_64-unknown-linux-gnu/bits/c++config.h" 2 3

// If platform uses neither visibility nor psuedo-visibility,

// specify empty default for namespace annotation macros.

// Certain function definitions that are meant to be overridable from

// user code are decorated with this macro. For some targets, this

// macro causes these definitions to be weak.

// The remainder of the prewritten config is automatic; all the

// user hooks are listed above.

// Create a boolean flag to be used to determine if --fast-math is set.

// This marks string literals in header files to be extracted for eventual

// translation. It is primarily used for messages in thrown exceptions; see

// src/functexcept.cc. We use \_\_N because the more traditional \_N is used

// for something else under certain OSes (see BADNAMES).

// For example, <windows.h> is known to #define min and max as macros...

// End of prewritten config; the settings discovered at configure time follow.

/\* config.h. Generated from config.h.in by configure. \*/

/\* config.h.in. Generated from configure.ac by autoheader. \*/

/\* Define to 1 if you have the `acosf' function. \*/

/\* Define to 1 if you have the `acosl' function. \*/

/\* Define to 1 if you have the `asinf' function. \*/

/\* Define to 1 if you have the `asinl' function. \*/

/\* Define to 1 if the target assembler supports .symver directive. \*/

/\* Define to 1 if you have the `atan2f' function. \*/

/\* Define to 1 if you have the `atan2l' function. \*/

/\* Define to 1 if you have the `atanf' function. \*/

/\* Define to 1 if you have the `atanl' function. \*/

/\* Define to 1 if the target assembler supports thread-local storage. \*/

/\* #undef \_GLIBCXX\_HAVE\_CC\_TLS \*/

/\* Define to 1 if you have the `ceilf' function. \*/

/\* Define to 1 if you have the `ceill' function. \*/

/\* Define to 1 if you have the <complex.h> header file. \*/

/\* Define to 1 if you have the `cosf' function. \*/

/\* Define to 1 if you have the `coshf' function. \*/

/\* Define to 1 if you have the `coshl' function. \*/

/\* Define to 1 if you have the `cosl' function. \*/

/\* Define to 1 if you have the <dlfcn.h> header file. \*/

/\* Define if EBADMSG exists. \*/

/\* Define if ECANCELED exists. \*/

/\* Define if EIDRM exists. \*/

/\* Define to 1 if you have the <endian.h> header file. \*/

/\* Define if ENODATA exists. \*/

/\* Define if ENOLINK exists. \*/

/\* Define if ENOSR exists. \*/

/\* Define if ENOSTR exists. \*/

/\* Define if ENOTRECOVERABLE exists. \*/

/\* Define if ENOTSUP exists. \*/

/\* Define if EOVERFLOW exists. \*/

/\* Define if EOWNERDEAD exists. \*/

/\* Define if EPROTO exists. \*/

/\* Define if ETIME exists. \*/

/\* Define if ETXTBSY exists. \*/

/\* Define to 1 if you have the <execinfo.h> header file. \*/

/\* Define to 1 if you have the `expf' function. \*/

/\* Define to 1 if you have the `expl' function. \*/

/\* Define to 1 if you have the `fabsf' function. \*/

/\* Define to 1 if you have the `fabsl' function. \*/

/\* Define to 1 if you have the <fenv.h> header file. \*/

/\* Define to 1 if you have the `finite' function. \*/

/\* Define to 1 if you have the `finitef' function. \*/

/\* Define to 1 if you have the `finitel' function. \*/

/\* Define to 1 if you have the <float.h> header file. \*/

/\* Define to 1 if you have the `floorf' function. \*/

/\* Define to 1 if you have the `floorl' function. \*/

/\* Define to 1 if you have the `fmodf' function. \*/

/\* Define to 1 if you have the `fmodl' function. \*/

/\* Define to 1 if you have the `fpclass' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_FPCLASS \*/

/\* Define to 1 if you have the <fp.h> header file. \*/

/\* #undef \_GLIBCXX\_HAVE\_FP\_H \*/

/\* Define to 1 if you have the `frexpf' function. \*/

/\* Define to 1 if you have the `frexpl' function. \*/

/\* Define if \_Unwind\_GetIPInfo is available. \*/

/\* Define if gthr-default.h exists (meaning that threading support is

enabled). \*/

/\* Define to 1 if you have the `hypot' function. \*/

/\* Define to 1 if you have the `hypotf' function. \*/

/\* Define to 1 if you have the `hypotl' function. \*/

/\* Define if you have the iconv() function. \*/

/\* Define to 1 if you have the <ieeefp.h> header file. \*/

/\* #undef \_GLIBCXX\_HAVE\_IEEEFP\_H \*/

/\* Define if int64\_t is available in <stdint.h>. \*/

/\* Define if int64\_t is a long. \*/

/\* Define if int64\_t is a long long. \*/

/\* #undef \_GLIBCXX\_HAVE\_INT64\_T\_LONG\_LONG \*/

/\* Define to 1 if you have the <inttypes.h> header file. \*/

/\* Define to 1 if you have the `isinf' function. \*/

/\* Define to 1 if you have the `isinff' function. \*/

/\* Define to 1 if you have the `isinfl' function. \*/

/\* Define to 1 if you have the `isnan' function. \*/

/\* Define to 1 if you have the `isnanf' function. \*/

/\* Define to 1 if you have the `isnanl' function. \*/

/\* Defined if iswblank exists. \*/

/\* Define if LC\_MESSAGES is available in <locale.h>. \*/

/\* Define to 1 if you have the `ldexpf' function. \*/

/\* Define to 1 if you have the `ldexpl' function. \*/

/\* Define to 1 if you have the <libintl.h> header file. \*/

/\* Only used in build directory testsuite\_hooks.h. \*/

/\* Only used in build directory testsuite\_hooks.h. \*/

/\* Only used in build directory testsuite\_hooks.h. \*/

/\* Only used in build directory testsuite\_hooks.h. \*/

/\* Only used in build directory testsuite\_hooks.h. \*/

/\* Define if futex syscall is available. \*/

/\* Define to 1 if you have the <locale.h> header file. \*/

/\* Define to 1 if you have the `log10f' function. \*/

/\* Define to 1 if you have the `log10l' function. \*/

/\* Define to 1 if you have the `logf' function. \*/

/\* Define to 1 if you have the `logl' function. \*/

/\* Define to 1 if you have the <machine/endian.h> header file. \*/

/\* #undef \_GLIBCXX\_HAVE\_MACHINE\_ENDIAN\_H \*/

/\* Define to 1 if you have the <machine/param.h> header file. \*/

/\* #undef \_GLIBCXX\_HAVE\_MACHINE\_PARAM\_H \*/

/\* Define if mbstate\_t exists in wchar.h. \*/

/\* Define to 1 if you have the <memory.h> header file. \*/

/\* Define to 1 if you have the `modf' function. \*/

/\* Define to 1 if you have the `modff' function. \*/

/\* Define to 1 if you have the `modfl' function. \*/

/\* Define to 1 if you have the <nan.h> header file. \*/

/\* #undef \_GLIBCXX\_HAVE\_NAN\_H \*/

/\* Define if poll is available in <poll.h>. \*/

/\* Define to 1 if you have the `powf' function. \*/

/\* Define to 1 if you have the `powl' function. \*/

/\* Define to 1 if you have the `qfpclass' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_QFPCLASS \*/

/\* Define to 1 if you have the `setenv' function. \*/

/\* Define to 1 if you have the `sincos' function. \*/

/\* Define to 1 if you have the `sincosf' function. \*/

/\* Define to 1 if you have the `sincosl' function. \*/

/\* Define to 1 if you have the `sinf' function. \*/

/\* Define to 1 if you have the `sinhf' function. \*/

/\* Define to 1 if you have the `sinhl' function. \*/

/\* Define to 1 if you have the `sinl' function. \*/

/\* Define to 1 if you have the `sqrtf' function. \*/

/\* Define to 1 if you have the `sqrtl' function. \*/

/\* Define to 1 if you have the <stdbool.h> header file. \*/

/\* Define to 1 if you have the <stdint.h> header file. \*/

/\* Define to 1 if you have the <stdlib.h> header file. \*/

/\* Define if strerror\_l is available in <string.h>. \*/

/\* #undef \_GLIBCXX\_HAVE\_STRERROR\_L \*/

/\* Define if strerror\_r is available in <string.h>. \*/

/\* Define to 1 if you have the <strings.h> header file. \*/

/\* Define to 1 if you have the <string.h> header file. \*/

/\* Define to 1 if you have the `strtof' function. \*/

/\* Define to 1 if you have the `strtold' function. \*/

/\* Define if strxfrm\_l is available in <string.h>. \*/

/\* Define to 1 if the target runtime linker supports binding the same symbol

to different versions. \*/

/\* Define to 1 if you have the <sys/filio.h> header file. \*/

/\* #undef \_GLIBCXX\_HAVE\_SYS\_FILIO\_H \*/

/\* Define to 1 if you have the <sys/ioctl.h> header file. \*/

/\* Define to 1 if you have the <sys/ipc.h> header file. \*/

/\* Define to 1 if you have the <sys/isa\_defs.h> header file. \*/

/\* #undef \_GLIBCXX\_HAVE\_SYS\_ISA\_DEFS\_H \*/

/\* Define to 1 if you have the <sys/machine.h> header file. \*/

/\* #undef \_GLIBCXX\_HAVE\_SYS\_MACHINE\_H \*/

/\* Define to 1 if you have the <sys/param.h> header file. \*/

/\* Define to 1 if you have the <sys/resource.h> header file. \*/

/\* Define to 1 if you have the <sys/sem.h> header file. \*/

/\* Define to 1 if you have the <sys/stat.h> header file. \*/

/\* Define to 1 if you have the <sys/time.h> header file. \*/

/\* Define to 1 if you have the <sys/types.h> header file. \*/

/\* Define to 1 if you have the <sys/uio.h> header file. \*/

/\* Define if S\_IFREG is available in <sys/stat.h>. \*/

/\* #undef \_GLIBCXX\_HAVE\_S\_IFREG \*/

/\* Define if S\_IFREG is available in <sys/stat.h>. \*/

/\* Define to 1 if you have the `tanf' function. \*/

/\* Define to 1 if you have the `tanhf' function. \*/

/\* Define to 1 if you have the `tanhl' function. \*/

/\* Define to 1 if you have the `tanl' function. \*/

/\* Define to 1 if you have the <tgmath.h> header file. \*/

/\* Define to 1 if the target supports thread-local storage. \*/

/\* Define to 1 if you have the <unistd.h> header file. \*/

/\* Defined if vfwscanf exists. \*/

/\* Defined if vswscanf exists. \*/

/\* Defined if vwscanf exists. \*/

/\* Define to 1 if you have the <wchar.h> header file. \*/

/\* Defined if wcstof exists. \*/

/\* Define to 1 if you have the <wctype.h> header file. \*/

/\* Define if writev is available in <sys/uio.h>. \*/

/\* Define to 1 if you have the `\_acosf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ACOSF \*/

/\* Define to 1 if you have the `\_acosl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ACOSL \*/

/\* Define to 1 if you have the `\_asinf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ASINF \*/

/\* Define to 1 if you have the `\_asinl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ASINL \*/

/\* Define to 1 if you have the `\_atan2f' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ATAN2F \*/

/\* Define to 1 if you have the `\_atan2l' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ATAN2L \*/

/\* Define to 1 if you have the `\_atanf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ATANF \*/

/\* Define to 1 if you have the `\_atanl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ATANL \*/

/\* Define to 1 if you have the `\_ceilf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_CEILF \*/

/\* Define to 1 if you have the `\_ceill' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_CEILL \*/

/\* Define to 1 if you have the `\_cosf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_COSF \*/

/\* Define to 1 if you have the `\_coshf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_COSHF \*/

/\* Define to 1 if you have the `\_coshl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_COSHL \*/

/\* Define to 1 if you have the `\_cosl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_COSL \*/

/\* Define to 1 if you have the `\_expf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_EXPF \*/

/\* Define to 1 if you have the `\_expl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_EXPL \*/

/\* Define to 1 if you have the `\_fabsf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FABSF \*/

/\* Define to 1 if you have the `\_fabsl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FABSL \*/

/\* Define to 1 if you have the `\_finite' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FINITE \*/

/\* Define to 1 if you have the `\_finitef' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FINITEF \*/

/\* Define to 1 if you have the `\_finitel' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FINITEL \*/

/\* Define to 1 if you have the `\_floorf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FLOORF \*/

/\* Define to 1 if you have the `\_floorl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FLOORL \*/

/\* Define to 1 if you have the `\_fmodf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FMODF \*/

/\* Define to 1 if you have the `\_fmodl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FMODL \*/

/\* Define to 1 if you have the `\_fpclass' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FPCLASS \*/

/\* Define to 1 if you have the `\_frexpf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FREXPF \*/

/\* Define to 1 if you have the `\_frexpl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_FREXPL \*/

/\* Define to 1 if you have the `\_hypot' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_HYPOT \*/

/\* Define to 1 if you have the `\_hypotf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_HYPOTF \*/

/\* Define to 1 if you have the `\_hypotl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_HYPOTL \*/

/\* Define to 1 if you have the `\_isinf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ISINF \*/

/\* Define to 1 if you have the `\_isinff' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ISINFF \*/

/\* Define to 1 if you have the `\_isinfl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ISINFL \*/

/\* Define to 1 if you have the `\_isnan' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ISNAN \*/

/\* Define to 1 if you have the `\_isnanf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ISNANF \*/

/\* Define to 1 if you have the `\_isnanl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_ISNANL \*/

/\* Define to 1 if you have the `\_ldexpf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_LDEXPF \*/

/\* Define to 1 if you have the `\_ldexpl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_LDEXPL \*/

/\* Define to 1 if you have the `\_log10f' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_LOG10F \*/

/\* Define to 1 if you have the `\_log10l' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_LOG10L \*/

/\* Define to 1 if you have the `\_logf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_LOGF \*/

/\* Define to 1 if you have the `\_logl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_LOGL \*/

/\* Define to 1 if you have the `\_modf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_MODF \*/

/\* Define to 1 if you have the `\_modff' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_MODFF \*/

/\* Define to 1 if you have the `\_modfl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_MODFL \*/

/\* Define to 1 if you have the `\_powf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_POWF \*/

/\* Define to 1 if you have the `\_powl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_POWL \*/

/\* Define to 1 if you have the `\_qfpclass' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_QFPCLASS \*/

/\* Define to 1 if you have the `\_sincos' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_SINCOS \*/

/\* Define to 1 if you have the `\_sincosf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_SINCOSF \*/

/\* Define to 1 if you have the `\_sincosl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_SINCOSL \*/

/\* Define to 1 if you have the `\_sinf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_SINF \*/

/\* Define to 1 if you have the `\_sinhf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_SINHF \*/

/\* Define to 1 if you have the `\_sinhl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_SINHL \*/

/\* Define to 1 if you have the `\_sinl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_SINL \*/

/\* Define to 1 if you have the `\_sqrtf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_SQRTF \*/

/\* Define to 1 if you have the `\_sqrtl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_SQRTL \*/

/\* Define to 1 if you have the `\_tanf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_TANF \*/

/\* Define to 1 if you have the `\_tanhf' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_TANHF \*/

/\* Define to 1 if you have the `\_tanhl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_TANHL \*/

/\* Define to 1 if you have the `\_tanl' function. \*/

/\* #undef \_GLIBCXX\_HAVE\_\_TANL \*/

/\* Define as const if the declaration of iconv() needs const. \*/

/\* Define to the sub-directory in which libtool stores uninstalled libraries.

\*/

/\* Name of package \*/

/\* #undef \_GLIBCXX\_PACKAGE \*/

/\* Define to the address where bug reports for this package should be sent. \*/

/\* Define to the full name of this package. \*/

/\* Define to the full name and version of this package. \*/

/\* Define to the one symbol short name of this package. \*/

/\* Define to the home page for this package. \*/

/\* Define to the version of this package. \*/

/\* The size of `char', as computed by sizeof. \*/

/\* #undef SIZEOF\_CHAR \*/

/\* The size of `int', as computed by sizeof. \*/

/\* #undef SIZEOF\_INT \*/

/\* The size of `long', as computed by sizeof. \*/

/\* #undef SIZEOF\_LONG \*/

/\* The size of `short', as computed by sizeof. \*/

/\* #undef SIZEOF\_SHORT \*/

/\* The size of `void \*', as computed by sizeof. \*/

/\* #undef SIZEOF\_VOID\_P \*/

/\* Define to 1 if you have the ANSI C header files. \*/

/\* Version number of package \*/

/\* #undef \_GLIBCXX\_VERSION \*/

/\* Define if builtin atomic operations for bool are supported on this host. \*/

/\* Define if builtin atomic operations for short are supported on this host.

\*/

/\* Define if builtin atomic operations for int are supported on this host. \*/

/\* Define if builtin atomic operations for long long are supported on this

host. \*/

/\* Define to use concept checking code from the boost libraries. \*/

/\* #undef \_GLIBCXX\_CONCEPT\_CHECKS \*/

/\* Define if a fully dynamic basic\_string is wanted. \*/

/\* #undef \_GLIBCXX\_FULLY\_DYNAMIC\_STRING \*/

/\* Define if gthreads library is available. \*/

/\* Define to 1 if a full hosted library is built, or 0 if freestanding. \*/

/\* Define if compatibility should be provided for -mlong-double-64. \*/

/\* Define if ptrdiff\_t is int. \*/

/\* #undef \_GLIBCXX\_PTRDIFF\_T\_IS\_INT \*/

/\* Define if using setrlimit to set resource limits during "make check" \*/

/\* Define if size\_t is unsigned int. \*/

/\* #undef \_GLIBCXX\_SIZE\_T\_IS\_UINT \*/

/\* Define if the compiler is configured for setjmp/longjmp exceptions. \*/

/\* #undef \_GLIBCXX\_SJLJ\_EXCEPTIONS \*/

/\* Define to the value of the EOF integer constant. \*/

/\* Define to the value of the SEEK\_CUR integer constant. \*/

/\* Define to the value of the SEEK\_END integer constant. \*/

/\* Define to use symbol versioning in the shared library. \*/

/\* Define to use darwin versioning in the shared library. \*/

/\* #undef \_GLIBCXX\_SYMVER\_DARWIN \*/

/\* Define to use GNU versioning in the shared library. \*/

/\* Define to use GNU namespace versioning in the shared library. \*/

/\* #undef \_GLIBCXX\_SYMVER\_GNU\_NAMESPACE \*/

/\* Define to use Sun versioning in the shared library. \*/

/\* #undef \_GLIBCXX\_SYMVER\_SUN \*/

/\* Define if C99 functions or macros from <wchar.h>, <math.h>, <complex.h>,

<stdio.h>, and <stdlib.h> can be used or exposed. \*/

/\* Define if C99 functions in <complex.h> should be used in <complex>. Using

compiler builtins for these functions requires corresponding C99 library

functions to be present. \*/

/\* Define if C99 functions in <complex.h> should be used in <tr1/complex>.

Using compiler builtins for these functions requires corresponding C99

library functions to be present. \*/

/\* Define if C99 functions in <ctype.h> should be imported in <tr1/cctype> in

namespace std::tr1. \*/

/\* Define if C99 functions in <fenv.h> should be imported in <tr1/cfenv> in

namespace std::tr1. \*/

/\* Define if C99 functions in <inttypes.h> should be imported in

<tr1/cinttypes> in namespace std::tr1. \*/

/\* Define if wchar\_t C99 functions in <inttypes.h> should be imported in

<tr1/cinttypes> in namespace std::tr1. \*/

/\* Define if C99 functions or macros in <math.h> should be imported in <cmath>

in namespace std. \*/

/\* Define if C99 functions or macros in <math.h> should be imported in

<tr1/cmath> in namespace std::tr1. \*/

/\* Define if C99 types in <stdint.h> should be imported in <tr1/cstdint> in

namespace std::tr1. \*/

/\* Defined if clock\_gettime has monotonic clock support. \*/

/\* #undef \_GLIBCXX\_USE\_CLOCK\_MONOTONIC \*/

/\* Defined if clock\_gettime has realtime clock support. \*/

/\* #undef \_GLIBCXX\_USE\_CLOCK\_REALTIME \*/

/\* Define if ISO/IEC TR 24733 decimal floating point types are supported on

this host. \*/

/\* Defined if gettimeofday is available. \*/

/\* Define if LFS support is available. \*/

/\* Define if code specialized for long long should be used. \*/

/\* Defined if nanosleep is available. \*/

/\* #undef \_GLIBCXX\_USE\_NANOSLEEP \*/

/\* Define if NLS translations are to be used. \*/

/\* Define if /dev/random and /dev/urandom are available for the random\_device

of TR1 (Chapter 5.1). \*/

/\* Defined if sched\_yield is available. \*/

/\* #undef \_GLIBCXX\_USE\_SCHED\_YIELD \*/

/\* Define if code specialized for wchar\_t should be used. \*/

#43 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/cmath" 2 3

#1 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/bits/cpp\_type\_traits.h" 1 3

// The -\*- C++ -\*- type traits classes for internal use in libstdc++

// Copyright (C) 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2009, 2010

// Free Software Foundation, Inc.

//

// This file is part of the GNU ISO C++ Library. This library is free

// software; you can redistribute it and/or modify it under the

// terms of the GNU General Public License as published by the

// Free Software Foundation; either version 3, or (at your option)

// any later version.

// This library is distributed in the hope that it will be useful,

// but WITHOUT ANY WARRANTY; without even the implied warranty of

// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

// GNU General Public License for more details.

// Under Section 7 of GPL version 3, you are granted additional

// permissions described in the GCC Runtime Library Exception, version

// 3.1, as published by the Free Software Foundation.

// You should have received a copy of the GNU General Public License and

// a copy of the GCC Runtime Library Exception along with this program;

// see the files COPYING3 and COPYING.RUNTIME respectively. If not, see

// <http://www.gnu.org/licenses/>.

/\*\* @file bits/cpp\_type\_traits.h

\* This is an internal header file, included by other library headers.

\* Do not attempt to use it directly. @headername{ext/type\_traits}

\*/

// Written by Gabriel Dos Reis <dosreis@cmla.ens-cachan.fr>

#36 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/bits/cpp\_type\_traits.h" 3

//

// This file provides some compile-time information about various types.

// These representations were designed, on purpose, to be constant-expressions

// and not types as found in <bits/type\_traits.h>. In particular, they

// can be used in control structures and the optimizer hopefully will do

// the obvious thing.

//

// Why integral expressions, and not functions nor types?

// Firstly, these compile-time entities are used as template-arguments

// so function return values won't work: We need compile-time entities.

// We're left with types and constant integral expressions.

// Secondly, from the point of view of ease of use, type-based compile-time

// information is -not- \*that\* convenient. On has to write lots of

// overloaded functions and to hope that the compiler will select the right

// one. As a net effect, the overall structure isn't very clear at first

// glance.

// Thirdly, partial ordering and overload resolution (of function templates)

// is highly costly in terms of compiler-resource. It is a Good Thing to

// keep these resource consumption as least as possible.

//

// See valarray\_array.h for a case use.

//

// -- Gaby (dosreis@cmla.ens-cachan.fr) 2000-03-06.

//

// Update 2005: types are also provided and <bits/type\_traits.h> has been

// removed.

//

// Forward declaration hack, should really include this from somewhere.

namespace \_\_gnu\_cxx \_\_attribute\_\_ ((\_\_visibility\_\_ ("default")))

{

template<typename \_Iterator, typename \_Container>

class \_\_normal\_iterator;

} // namespace

namespace std \_\_attribute\_\_ ((\_\_visibility\_\_ ("default")))

{

struct \_\_true\_type { };

struct \_\_false\_type { };

template<bool>

struct \_\_truth\_type

{ typedef \_\_false\_type \_\_type; };

template<>

struct \_\_truth\_type<true>

{ typedef \_\_true\_type \_\_type; };

// N.B. The conversions to bool are needed due to the issue

// explained in c++/19404.

template<class \_Sp, class \_Tp>

struct \_\_traitor

{

enum { \_\_value = bool(\_Sp::\_\_value) || bool(\_Tp::\_\_value) };

typedef typename \_\_truth\_type<\_\_value>::\_\_type \_\_type;

};

// Compare for equality of types.

template<typename, typename>

struct \_\_are\_same

{

enum { \_\_value = 0 };

typedef \_\_false\_type \_\_type;

};

template<typename \_Tp>

struct \_\_are\_same<\_Tp, \_Tp>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

// Holds if the template-argument is a void type.

template<typename \_Tp>

struct \_\_is\_void

{

enum { \_\_value = 0 };

typedef \_\_false\_type \_\_type;

};

template<>

struct \_\_is\_void<void>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

//

// Integer types

//

template<typename \_Tp>

struct \_\_is\_integer

{

enum { \_\_value = 0 };

typedef \_\_false\_type \_\_type;

};

// Thirteen specializations (yes there are eleven standard integer

// types; <em>long long</em> and <em>unsigned long long</em> are

// supported as extensions)

template<>

struct \_\_is\_integer<bool>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_integer<char>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_integer<signed char>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_integer<unsigned char>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_integer<wchar\_t>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

#198 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/bits/cpp\_type\_traits.h" 3

template<>

struct \_\_is\_integer<short>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_integer<unsigned short>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_integer<int>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_integer<unsigned int>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_integer<long>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_integer<unsigned long>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_integer<long long>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_integer<unsigned long long>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

//

// Floating point types

//

template<typename \_Tp>

struct \_\_is\_floating

{

enum { \_\_value = 0 };

typedef \_\_false\_type \_\_type;

};

// three specializations (float, double and 'long double')

template<>

struct \_\_is\_floating<float>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_floating<double>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_floating<long double>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

//

// Pointer types

//

template<typename \_Tp>

struct \_\_is\_pointer

{

enum { \_\_value = 0 };

typedef \_\_false\_type \_\_type;

};

template<typename \_Tp>

struct \_\_is\_pointer<\_Tp\*>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

//

// Normal iterator type

//

template<typename \_Tp>

struct \_\_is\_normal\_iterator

{

enum { \_\_value = 0 };

typedef \_\_false\_type \_\_type;

};

template<typename \_Iterator, typename \_Container>

struct \_\_is\_normal\_iterator< \_\_gnu\_cxx::\_\_normal\_iterator<\_Iterator,

\_Container> >

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

//

// An arithmetic type is an integer type or a floating point type

//

template<typename \_Tp>

struct \_\_is\_arithmetic

: public \_\_traitor<\_\_is\_integer<\_Tp>, \_\_is\_floating<\_Tp> >

{ };

//

// A fundamental type is `void' or and arithmetic type

//

template<typename \_Tp>

struct \_\_is\_fundamental

: public \_\_traitor<\_\_is\_void<\_Tp>, \_\_is\_arithmetic<\_Tp> >

{ };

//

// A scalar type is an arithmetic type or a pointer type

//

template<typename \_Tp>

struct \_\_is\_scalar

: public \_\_traitor<\_\_is\_arithmetic<\_Tp>, \_\_is\_pointer<\_Tp> >

{ };

//

// For use in std::copy and std::find overloads for streambuf iterators.

//

template<typename \_Tp>

struct \_\_is\_char

{

enum { \_\_value = 0 };

typedef \_\_false\_type \_\_type;

};

template<>

struct \_\_is\_char<char>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_char<wchar\_t>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<typename \_Tp>

struct \_\_is\_byte

{

enum { \_\_value = 0 };

typedef \_\_false\_type \_\_type;

};

template<>

struct \_\_is\_byte<char>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_byte<signed char>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

template<>

struct \_\_is\_byte<unsigned char>

{

enum { \_\_value = 1 };

typedef \_\_true\_type \_\_type;

};

//

// Move iterator type

//

template<typename \_Tp>

struct \_\_is\_move\_iterator

{

enum { \_\_value = 0 };

typedef \_\_false\_type \_\_type;

};

#422 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/bits/cpp\_type\_traits.h" 3

} // namespace

#44 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/cmath" 2 3

#1 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/ext/type\_traits.h" 1 3

// -\*- C++ -\*-

// Copyright (C) 2005, 2006, 2007, 2009, 2010, 2011

// Free Software Foundation, Inc.

//

// This file is part of the GNU ISO C++ Library. This library is free

// software; you can redistribute it and/or modify it under the terms

// of the GNU General Public License as published by the Free Software

// Foundation; either version 3, or (at your option) any later

// version.

// This library is distributed in the hope that it will be useful, but

// WITHOUT ANY WARRANTY; without even the implied warranty of

// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

// General Public License for more details.

// Under Section 7 of GPL version 3, you are granted additional

// permissions described in the GCC Runtime Library Exception, version

// 3.1, as published by the Free Software Foundation.

// You should have received a copy of the GNU General Public License and

// a copy of the GCC Runtime Library Exception along with this program;

// see the files COPYING3 and COPYING.RUNTIME respectively. If not, see

// <http://www.gnu.org/licenses/>.

/\*\* @file ext/type\_traits.h

\* This file is a GNU extension to the Standard C++ Library.

\*/

#33 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/ext/type\_traits.h" 3

namespace \_\_gnu\_cxx \_\_attribute\_\_ ((\_\_visibility\_\_ ("default")))

{

// Define a nested type if some predicate holds.

template<bool, typename>

struct \_\_enable\_if

{ };

template<typename \_Tp>

struct \_\_enable\_if<true, \_Tp>

{ typedef \_Tp \_\_type; };

// Conditional expression for types. If true, first, if false, second.

template<bool \_Cond, typename \_Iftrue, typename \_Iffalse>

struct \_\_conditional\_type

{ typedef \_Iftrue \_\_type; };

template<typename \_Iftrue, typename \_Iffalse>

struct \_\_conditional\_type<false, \_Iftrue, \_Iffalse>

{ typedef \_Iffalse \_\_type; };

// Given an integral builtin type, return the corresponding unsigned type.

template<typename \_Tp>

struct \_\_add\_unsigned

{

private:

typedef \_\_enable\_if<std::\_\_is\_integer<\_Tp>::\_\_value, \_Tp> \_\_if\_type;

public:

typedef typename \_\_if\_type::\_\_type \_\_type;

};

template<>

struct \_\_add\_unsigned<char>

{ typedef unsigned char \_\_type; };

template<>

struct \_\_add\_unsigned<signed char>

{ typedef unsigned char \_\_type; };

template<>

struct \_\_add\_unsigned<short>

{ typedef unsigned short \_\_type; };

template<>

struct \_\_add\_unsigned<int>

{ typedef unsigned int \_\_type; };

template<>

struct \_\_add\_unsigned<long>

{ typedef unsigned long \_\_type; };

template<>

struct \_\_add\_unsigned<long long>

{ typedef unsigned long long \_\_type; };

// Declare but don't define.

template<>

struct \_\_add\_unsigned<bool>;

template<>

struct \_\_add\_unsigned<wchar\_t>;

// Given an integral builtin type, return the corresponding signed type.

template<typename \_Tp>

struct \_\_remove\_unsigned

{

private:

typedef \_\_enable\_if<std::\_\_is\_integer<\_Tp>::\_\_value, \_Tp> \_\_if\_type;

public:

typedef typename \_\_if\_type::\_\_type \_\_type;

};

template<>

struct \_\_remove\_unsigned<char>

{ typedef signed char \_\_type; };

template<>

struct \_\_remove\_unsigned<unsigned char>

{ typedef signed char \_\_type; };

template<>

struct \_\_remove\_unsigned<unsigned short>

{ typedef short \_\_type; };

template<>

struct \_\_remove\_unsigned<unsigned int>

{ typedef int \_\_type; };

template<>

struct \_\_remove\_unsigned<unsigned long>

{ typedef long \_\_type; };

template<>

struct \_\_remove\_unsigned<unsigned long long>

{ typedef long long \_\_type; };

// Declare but don't define.

template<>

struct \_\_remove\_unsigned<bool>;

template<>

struct \_\_remove\_unsigned<wchar\_t>;

// For use in string and vstring.

template<typename \_Type>

inline bool

\_\_is\_null\_pointer(\_Type\* \_\_ptr)

{ return \_\_ptr == 0; }

template<typename \_Type>

inline bool

\_\_is\_null\_pointer(\_Type)

{ return false; }

// For complex and cmath

template<typename \_Tp, bool = std::\_\_is\_integer<\_Tp>::\_\_value>

struct \_\_promote

{ typedef double \_\_type; };

// No nested \_\_type member for non-integer non-floating point types,

// allows this type to be used for SFINAE to constrain overloads in

// <cmath> and <complex> to only the intended types.

template<typename \_Tp>

struct \_\_promote<\_Tp, false>

{ };

template<>

struct \_\_promote<long double>

{ typedef long double \_\_type; };

template<>

struct \_\_promote<double>

{ typedef double \_\_type; };

template<>

struct \_\_promote<float>

{ typedef float \_\_type; };

template<typename \_Tp, typename \_Up,

typename \_Tp2 = typename \_\_promote<\_Tp>::\_\_type,

typename \_Up2 = typename \_\_promote<\_Up>::\_\_type>

struct \_\_promote\_2

{

typedef \_\_typeof\_\_(\_Tp2() + \_Up2()) \_\_type;

};

template<typename \_Tp, typename \_Up, typename \_Vp,

typename \_Tp2 = typename \_\_promote<\_Tp>::\_\_type,

typename \_Up2 = typename \_\_promote<\_Up>::\_\_type,

typename \_Vp2 = typename \_\_promote<\_Vp>::\_\_type>

struct \_\_promote\_3

{

typedef \_\_typeof\_\_(\_Tp2() + \_Up2() + \_Vp2()) \_\_type;

};

template<typename \_Tp, typename \_Up, typename \_Vp, typename \_Wp,

typename \_Tp2 = typename \_\_promote<\_Tp>::\_\_type,

typename \_Up2 = typename \_\_promote<\_Up>::\_\_type,

typename \_Vp2 = typename \_\_promote<\_Vp>::\_\_type,

typename \_Wp2 = typename \_\_promote<\_Wp>::\_\_type>

struct \_\_promote\_4

{

typedef \_\_typeof\_\_(\_Tp2() + \_Up2() + \_Vp2() + \_Wp2()) \_\_type;

};

} // namespace

#45 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/cmath" 2 3

extern "C" {

#1 "/usr/include/math.h" 1 3 4

/\* Declarations for math functions.

Copyright (C) 1991-1993, 1995-1999, 2001, 2002, 2004, 2006, 2009, 2011

Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

/\*

\* ISO C99 Standard: 7.12 Mathematics <math.h>

\*/

extern "C" {

/\* Get machine-dependent HUGE\_VAL value (returned on overflow).

On all IEEE754 machines, this is +Infinity. \*/

#1 "/usr/include/x86\_64-linux-gnu/bits/huge\_val.h" 1 3 4

/\* `HUGE\_VAL' constant for IEEE 754 machines (where it is infinity).

Used by <stdlib.h> and <math.h> functions for overflow.

Copyright (C) 1992, 1995, 1996, 1997, 1999, 2000, 2004

Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

/\* IEEE positive infinity (-HUGE\_VAL is negative infinity). \*/

#35 "/usr/include/math.h" 2 3 4

#1 "/usr/include/x86\_64-linux-gnu/bits/huge\_valf.h" 1 3 4

/\* `HUGE\_VALF' constant for IEEE 754 machines (where it is infinity).

Used by <stdlib.h> and <math.h> functions for overflow.

Copyright (C) 1992, 1995, 1996, 1997, 1999, 2000, 2004

Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

/\* IEEE positive infinity (-HUGE\_VAL is negative infinity). \*/

#37 "/usr/include/math.h" 2 3 4

#1 "/usr/include/x86\_64-linux-gnu/bits/huge\_vall.h" 1 3 4

/\* Default `HUGE\_VALL' constant.

Used by <stdlib.h> and <math.h> functions for overflow.

Copyright (C) 1992, 1996, 1997, 2004 Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

#38 "/usr/include/math.h" 2 3 4

/\* Get machine-dependent INFINITY value. \*/

#1 "/usr/include/x86\_64-linux-gnu/bits/inf.h" 1 3 4

/\* `INFINITY' constant for IEEE 754 machines.

Copyright (C) 2004 Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

/\* IEEE positive infinity. \*/

#41 "/usr/include/math.h" 2 3 4

/\* Get machine-dependent NAN value (returned for some domain errors). \*/

#1 "/usr/include/x86\_64-linux-gnu/bits/nan.h" 1 3 4

/\* `NAN' constant for IEEE 754 machines.

Copyright (C) 1992,1996,1997,1999,2004,2006 Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

/\* IEEE Not A Number. \*/

#44 "/usr/include/math.h" 2 3 4

/\* Get general and ISO C99 specific information. \*/

#1 "/usr/include/x86\_64-linux-gnu/bits/mathdef.h" 1 3 4

/\* Copyright (C) 2001, 2004, 2010 Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

#26 "/usr/include/x86\_64-linux-gnu/bits/mathdef.h" 3 4

#1 "/usr/include/x86\_64-linux-gnu/bits/wordsize.h" 1 3 4

/\* Determine the wordsize from the preprocessor defines. \*/

#27 "/usr/include/x86\_64-linux-gnu/bits/mathdef.h" 2 3 4

/\* The x86-64 architecture computes values with the precission of the

used type. Similarly for -m32 -mfpmath=sse. \*/

typedef float float\_t; /\* `float' expressions are evaluated as `float'. \*/

typedef double double\_t; /\* `double' expressions are evaluated

as `double'. \*/

#44 "/usr/include/x86\_64-linux-gnu/bits/mathdef.h" 3 4

/\* The values returned by `ilogb' for 0 and NaN respectively. \*/

/\* The GCC 4.6 compiler will define \_\_FP\_FAST\_FMA{,F,L} if the fma{,f,l}

builtins are supported. \*/

#48 "/usr/include/math.h" 2 3 4

/\* The file <bits/mathcalls.h> contains the prototypes for all the

actual math functions. These macros are used for those prototypes,

so we can easily declare each function as both `name' and `\_\_name',

and can declare the float versions `namef' and `\_\_namef'. \*/

#71 "/usr/include/math.h" 3 4

#1 "/usr/include/x86\_64-linux-gnu/bits/mathcalls.h" 1 3 4

/\* Prototype declarations for math functions; helper file for <math.h>.

Copyright (C) 1996-2002, 2003, 2006, 2011 Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

/\* NOTE: Because of the special way this file is used by <math.h>, this

file must NOT be protected from multiple inclusion as header files

usually are.

This file provides prototype declarations for the math functions.

Most functions are declared using the macro:

\_\_MATHCALL (NAME,[\_r], (ARGS...));

This means there is a function `NAME' returning `double' and a function

`NAMEf' returning `float'. Each place `\_Mdouble\_' appears in the

prototype, that is actually `double' in the prototype for `NAME' and

`float' in the prototype for `NAMEf'. Reentrant variant functions are

called `NAME\_r' and `NAMEf\_r'.

Functions returning other types like `int' are declared using the macro:

\_\_MATHDECL (TYPE, NAME,[\_r], (ARGS...));

This is just like \_\_MATHCALL but for a function returning `TYPE'

instead of `\_Mdouble\_'. In all of these cases, there is still

both a `NAME' and a `NAMEf' that takes `float' arguments.

Note that there must be no whitespace before the argument passed for

NAME, to make token pasting work with -traditional. \*/

/\* Trigonometric functions. \*/

/\* Arc cosine of X. \*/

extern double acos (double \_\_x) throw (); extern double \_\_acos (double \_\_x) throw ();

/\* Arc sine of X. \*/

extern double asin (double \_\_x) throw (); extern double \_\_asin (double \_\_x) throw ();

/\* Arc tangent of X. \*/

extern double atan (double \_\_x) throw (); extern double \_\_atan (double \_\_x) throw ();

/\* Arc tangent of Y/X. \*/

extern double atan2 (double \_\_y, double \_\_x) throw (); extern double \_\_atan2 (double \_\_y, double \_\_x) throw ();

/\* Cosine of X. \*/

extern double cos (double \_\_x) throw (); extern double \_\_cos (double \_\_x) throw ();

/\* Sine of X. \*/

extern double sin (double \_\_x) throw (); extern double \_\_sin (double \_\_x) throw ();

/\* Tangent of X. \*/

extern double tan (double \_\_x) throw (); extern double \_\_tan (double \_\_x) throw ();

/\* Hyperbolic functions. \*/

/\* Hyperbolic cosine of X. \*/

extern double cosh (double \_\_x) throw (); extern double \_\_cosh (double \_\_x) throw ();

/\* Hyperbolic sine of X. \*/

extern double sinh (double \_\_x) throw (); extern double \_\_sinh (double \_\_x) throw ();

/\* Hyperbolic tangent of X. \*/

extern double tanh (double \_\_x) throw (); extern double \_\_tanh (double \_\_x) throw ();

/\* Cosine and sine of X. \*/

extern void sincos (double \_\_x, double \*\_\_sinx, double \*\_\_cosx) throw (); extern void \_\_sincos (double \_\_x, double \*\_\_sinx, double \*\_\_cosx) throw ();

/\* Hyperbolic arc cosine of X. \*/

extern double acosh (double \_\_x) throw (); extern double \_\_acosh (double \_\_x) throw ();

/\* Hyperbolic arc sine of X. \*/

extern double asinh (double \_\_x) throw (); extern double \_\_asinh (double \_\_x) throw ();

/\* Hyperbolic arc tangent of X. \*/

extern double atanh (double \_\_x) throw (); extern double \_\_atanh (double \_\_x) throw ();

/\* Exponential and logarithmic functions. \*/

/\* Exponential function of X. \*/

extern double exp (double \_\_x) throw (); extern double \_\_exp (double \_\_x) throw ();

/\* Break VALUE into a normalized fraction and an integral power of 2. \*/

extern double frexp (double \_\_x, int \*\_\_exponent) throw (); extern double \_\_frexp (double \_\_x, int \*\_\_exponent) throw ();

/\* X times (two to the EXP power). \*/

extern double ldexp (double \_\_x, int \_\_exponent) throw (); extern double \_\_ldexp (double \_\_x, int \_\_exponent) throw ();

/\* Natural logarithm of X. \*/

extern double log (double \_\_x) throw (); extern double \_\_log (double \_\_x) throw ();

/\* Base-ten logarithm of X. \*/

extern double log10 (double \_\_x) throw (); extern double \_\_log10 (double \_\_x) throw ();

/\* Break VALUE into integral and fractional parts. \*/

extern double modf (double \_\_x, double \*\_\_iptr) throw (); extern double \_\_modf (double \_\_x, double \*\_\_iptr) throw ()

\_\_attribute\_\_ ((\_\_nonnull\_\_ (2)));

/\* A function missing in all standards: compute exponent to base ten. \*/

extern double exp10 (double \_\_x) throw (); extern double \_\_exp10 (double \_\_x) throw ();

/\* Another name occasionally used. \*/

extern double pow10 (double \_\_x) throw (); extern double \_\_pow10 (double \_\_x) throw ();

/\* Return exp(X) - 1. \*/

extern double expm1 (double \_\_x) throw (); extern double \_\_expm1 (double \_\_x) throw ();

/\* Return log(1 + X). \*/

extern double log1p (double \_\_x) throw (); extern double \_\_log1p (double \_\_x) throw ();

/\* Return the base 2 signed integral exponent of X. \*/

extern double logb (double \_\_x) throw (); extern double \_\_logb (double \_\_x) throw ();

/\* Compute base-2 exponential of X. \*/

extern double exp2 (double \_\_x) throw (); extern double \_\_exp2 (double \_\_x) throw ();

/\* Compute base-2 logarithm of X. \*/

extern double log2 (double \_\_x) throw (); extern double \_\_log2 (double \_\_x) throw ();

/\* Power functions. \*/

/\* Return X to the Y power. \*/

extern double pow (double \_\_x, double \_\_y) throw (); extern double \_\_pow (double \_\_x, double \_\_y) throw ();

/\* Return the square root of X. \*/

extern double sqrt (double \_\_x) throw (); extern double \_\_sqrt (double \_\_x) throw ();

/\* Return `sqrt(X\*X + Y\*Y)'. \*/

extern double hypot (double \_\_x, double \_\_y) throw (); extern double \_\_hypot (double \_\_x, double \_\_y) throw ();

/\* Return the cube root of X. \*/

extern double cbrt (double \_\_x) throw (); extern double \_\_cbrt (double \_\_x) throw ();

/\* Nearest integer, absolute value, and remainder functions. \*/

/\* Smallest integral value not less than X. \*/

extern double ceil (double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern double \_\_ceil (double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Absolute value of X. \*/

extern double fabs (double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern double \_\_fabs (double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Largest integer not greater than X. \*/

extern double floor (double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern double \_\_floor (double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Floating-point modulo remainder of X/Y. \*/

extern double fmod (double \_\_x, double \_\_y) throw (); extern double \_\_fmod (double \_\_x, double \_\_y) throw ();

/\* Return 0 if VALUE is finite or NaN, +1 if it

is +Infinity, -1 if it is -Infinity. \*/

extern int \_\_isinf (double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is finite and not NaN. \*/

extern int \_\_finite (double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return 0 if VALUE is finite or NaN, +1 if it

is +Infinity, -1 if it is -Infinity. \*/

extern int isinf (double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is finite and not NaN. \*/

extern int finite (double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return the remainder of X/Y. \*/

extern double drem (double \_\_x, double \_\_y) throw (); extern double \_\_drem (double \_\_x, double \_\_y) throw ();

/\* Return the fractional part of X after dividing out `ilogb (X)'. \*/

extern double significand (double \_\_x) throw (); extern double \_\_significand (double \_\_x) throw ();

/\* Return X with its signed changed to Y's. \*/

extern double copysign (double \_\_x, double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern double \_\_copysign (double \_\_x, double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return representation of NaN for double type. \*/

extern double nan (\_\_const char \*\_\_tagb) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern double \_\_nan (\_\_const char \*\_\_tagb) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is not a number. \*/

extern int \_\_isnan (double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is not a number. \*/

extern int isnan (double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Bessel functions. \*/

extern double j0 (double) throw (); extern double \_\_j0 (double) throw ();

extern double j1 (double) throw (); extern double \_\_j1 (double) throw ();

extern double jn (int, double) throw (); extern double \_\_jn (int, double) throw ();

extern double y0 (double) throw (); extern double \_\_y0 (double) throw ();

extern double y1 (double) throw (); extern double \_\_y1 (double) throw ();

extern double yn (int, double) throw (); extern double \_\_yn (int, double) throw ();

/\* Error and gamma functions. \*/

extern double erf (double) throw (); extern double \_\_erf (double) throw ();

extern double erfc (double) throw (); extern double \_\_erfc (double) throw ();

extern double lgamma (double) throw (); extern double \_\_lgamma (double) throw ();

/\* True gamma function. \*/

extern double tgamma (double) throw (); extern double \_\_tgamma (double) throw ();

/\* Obsolete alias for `lgamma'. \*/

extern double gamma (double) throw (); extern double \_\_gamma (double) throw ();

/\* Reentrant version of lgamma. This function uses the global variable

`signgam'. The reentrant version instead takes a pointer and stores

the value through it. \*/

extern double lgamma\_r (double, int \*\_\_signgamp) throw (); extern double \_\_lgamma\_r (double, int \*\_\_signgamp) throw ();

/\* Return the integer nearest X in the direction of the

prevailing rounding mode. \*/

extern double rint (double \_\_x) throw (); extern double \_\_rint (double \_\_x) throw ();

/\* Return X + epsilon if X < Y, X - epsilon if X > Y. \*/

extern double nextafter (double \_\_x, double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern double \_\_nextafter (double \_\_x, double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_));

extern double nexttoward (double \_\_x, long double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern double \_\_nexttoward (double \_\_x, long double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return the remainder of integer divison X / Y with infinite precision. \*/

extern double remainder (double \_\_x, double \_\_y) throw (); extern double \_\_remainder (double \_\_x, double \_\_y) throw ();

/\* Return X times (2 to the Nth power). \*/

extern double scalbn (double \_\_x, int \_\_n) throw (); extern double \_\_scalbn (double \_\_x, int \_\_n) throw ();

/\* Return the binary exponent of X, which must be nonzero. \*/

extern int ilogb (double \_\_x) throw (); extern int \_\_ilogb (double \_\_x) throw ();

/\* Return X times (2 to the Nth power). \*/

extern double scalbln (double \_\_x, long int \_\_n) throw (); extern double \_\_scalbln (double \_\_x, long int \_\_n) throw ();

/\* Round X to integral value in floating-point format using current

rounding direction, but do not raise inexact exception. \*/

extern double nearbyint (double \_\_x) throw (); extern double \_\_nearbyint (double \_\_x) throw ();

/\* Round X to nearest integral value, rounding halfway cases away from

zero. \*/

extern double round (double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern double \_\_round (double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Round X to the integral value in floating-point format nearest but

not larger in magnitude. \*/

extern double trunc (double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern double \_\_trunc (double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Compute remainder of X and Y and put in \*QUO a value with sign of x/y

and magnitude congruent `mod 2^n' to the magnitude of the integral

quotient x/y, with n >= 3. \*/

extern double remquo (double \_\_x, double \_\_y, int \*\_\_quo) throw (); extern double \_\_remquo (double \_\_x, double \_\_y, int \*\_\_quo) throw ();

/\* Conversion functions. \*/

/\* Round X to nearest integral value according to current rounding

direction. \*/

extern long int lrint (double \_\_x) throw (); extern long int \_\_lrint (double \_\_x) throw ();

extern long long int llrint (double \_\_x) throw (); extern long long int \_\_llrint (double \_\_x) throw ();

/\* Round X to nearest integral value, rounding halfway cases away from

zero. \*/

extern long int lround (double \_\_x) throw (); extern long int \_\_lround (double \_\_x) throw ();

extern long long int llround (double \_\_x) throw (); extern long long int \_\_llround (double \_\_x) throw ();

/\* Return positive difference between X and Y. \*/

extern double fdim (double \_\_x, double \_\_y) throw (); extern double \_\_fdim (double \_\_x, double \_\_y) throw ();

/\* Return maximum numeric value from X and Y. \*/

extern double fmax (double \_\_x, double \_\_y) throw (); extern double \_\_fmax (double \_\_x, double \_\_y) throw ();

/\* Return minimum numeric value from X and Y. \*/

extern double fmin (double \_\_x, double \_\_y) throw (); extern double \_\_fmin (double \_\_x, double \_\_y) throw ();

/\* Classify given number. \*/

extern int \_\_fpclassify (double \_\_value) throw ()

\_\_attribute\_\_ ((\_\_const\_\_));

/\* Test for negative number. \*/

extern int \_\_signbit (double \_\_value) throw ()

\_\_attribute\_\_ ((\_\_const\_\_));

/\* Multiply-add function computed as a ternary operation. \*/

extern double fma (double \_\_x, double \_\_y, double \_\_z) throw (); extern double \_\_fma (double \_\_x, double \_\_y, double \_\_z) throw ();

/\* Return X times (2 to the Nth power). \*/

extern double scalb (double \_\_x, double \_\_n) throw (); extern double \_\_scalb (double \_\_x, double \_\_n) throw ();

#72 "/usr/include/math.h" 2 3 4

/\* Include the file of declarations again, this time using `float'

instead of `double' and appending f to each function name. \*/

#94 "/usr/include/math.h" 3 4

#1 "/usr/include/x86\_64-linux-gnu/bits/mathcalls.h" 1 3 4

/\* Prototype declarations for math functions; helper file for <math.h>.

Copyright (C) 1996-2002, 2003, 2006, 2011 Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

/\* NOTE: Because of the special way this file is used by <math.h>, this

file must NOT be protected from multiple inclusion as header files

usually are.

This file provides prototype declarations for the math functions.

Most functions are declared using the macro:

\_\_MATHCALL (NAME,[\_r], (ARGS...));

This means there is a function `NAME' returning `double' and a function

`NAMEf' returning `float'. Each place `\_Mdouble\_' appears in the

prototype, that is actually `double' in the prototype for `NAME' and

`float' in the prototype for `NAMEf'. Reentrant variant functions are

called `NAME\_r' and `NAMEf\_r'.

Functions returning other types like `int' are declared using the macro:

\_\_MATHDECL (TYPE, NAME,[\_r], (ARGS...));

This is just like \_\_MATHCALL but for a function returning `TYPE'

instead of `\_Mdouble\_'. In all of these cases, there is still

both a `NAME' and a `NAMEf' that takes `float' arguments.

Note that there must be no whitespace before the argument passed for

NAME, to make token pasting work with -traditional. \*/

/\* Trigonometric functions. \*/

/\* Arc cosine of X. \*/

extern float acosf (float \_\_x) throw (); extern float \_\_acosf (float \_\_x) throw ();

/\* Arc sine of X. \*/

extern float asinf (float \_\_x) throw (); extern float \_\_asinf (float \_\_x) throw ();

/\* Arc tangent of X. \*/

extern float atanf (float \_\_x) throw (); extern float \_\_atanf (float \_\_x) throw ();

/\* Arc tangent of Y/X. \*/

extern float atan2f (float \_\_y, float \_\_x) throw (); extern float \_\_atan2f (float \_\_y, float \_\_x) throw ();

/\* Cosine of X. \*/

extern float cosf (float \_\_x) throw (); extern float \_\_cosf (float \_\_x) throw ();

/\* Sine of X. \*/

extern float sinf (float \_\_x) throw (); extern float \_\_sinf (float \_\_x) throw ();

/\* Tangent of X. \*/

extern float tanf (float \_\_x) throw (); extern float \_\_tanf (float \_\_x) throw ();

/\* Hyperbolic functions. \*/

/\* Hyperbolic cosine of X. \*/

extern float coshf (float \_\_x) throw (); extern float \_\_coshf (float \_\_x) throw ();

/\* Hyperbolic sine of X. \*/

extern float sinhf (float \_\_x) throw (); extern float \_\_sinhf (float \_\_x) throw ();

/\* Hyperbolic tangent of X. \*/

extern float tanhf (float \_\_x) throw (); extern float \_\_tanhf (float \_\_x) throw ();

/\* Cosine and sine of X. \*/

extern void sincosf (float \_\_x, float \*\_\_sinx, float \*\_\_cosx) throw (); extern void \_\_sincosf (float \_\_x, float \*\_\_sinx, float \*\_\_cosx) throw ();

/\* Hyperbolic arc cosine of X. \*/

extern float acoshf (float \_\_x) throw (); extern float \_\_acoshf (float \_\_x) throw ();

/\* Hyperbolic arc sine of X. \*/

extern float asinhf (float \_\_x) throw (); extern float \_\_asinhf (float \_\_x) throw ();

/\* Hyperbolic arc tangent of X. \*/

extern float atanhf (float \_\_x) throw (); extern float \_\_atanhf (float \_\_x) throw ();

/\* Exponential and logarithmic functions. \*/

/\* Exponential function of X. \*/

extern float expf (float \_\_x) throw (); extern float \_\_expf (float \_\_x) throw ();

/\* Break VALUE into a normalized fraction and an integral power of 2. \*/

extern float frexpf (float \_\_x, int \*\_\_exponent) throw (); extern float \_\_frexpf (float \_\_x, int \*\_\_exponent) throw ();

/\* X times (two to the EXP power). \*/

extern float ldexpf (float \_\_x, int \_\_exponent) throw (); extern float \_\_ldexpf (float \_\_x, int \_\_exponent) throw ();

/\* Natural logarithm of X. \*/

extern float logf (float \_\_x) throw (); extern float \_\_logf (float \_\_x) throw ();

/\* Base-ten logarithm of X. \*/

extern float log10f (float \_\_x) throw (); extern float \_\_log10f (float \_\_x) throw ();

/\* Break VALUE into integral and fractional parts. \*/

extern float modff (float \_\_x, float \*\_\_iptr) throw (); extern float \_\_modff (float \_\_x, float \*\_\_iptr) throw ()

\_\_attribute\_\_ ((\_\_nonnull\_\_ (2)));

/\* A function missing in all standards: compute exponent to base ten. \*/

extern float exp10f (float \_\_x) throw (); extern float \_\_exp10f (float \_\_x) throw ();

/\* Another name occasionally used. \*/

extern float pow10f (float \_\_x) throw (); extern float \_\_pow10f (float \_\_x) throw ();

/\* Return exp(X) - 1. \*/

extern float expm1f (float \_\_x) throw (); extern float \_\_expm1f (float \_\_x) throw ();

/\* Return log(1 + X). \*/

extern float log1pf (float \_\_x) throw (); extern float \_\_log1pf (float \_\_x) throw ();

/\* Return the base 2 signed integral exponent of X. \*/

extern float logbf (float \_\_x) throw (); extern float \_\_logbf (float \_\_x) throw ();

/\* Compute base-2 exponential of X. \*/

extern float exp2f (float \_\_x) throw (); extern float \_\_exp2f (float \_\_x) throw ();

/\* Compute base-2 logarithm of X. \*/

extern float log2f (float \_\_x) throw (); extern float \_\_log2f (float \_\_x) throw ();

/\* Power functions. \*/

/\* Return X to the Y power. \*/

extern float powf (float \_\_x, float \_\_y) throw (); extern float \_\_powf (float \_\_x, float \_\_y) throw ();

/\* Return the square root of X. \*/

extern float sqrtf (float \_\_x) throw (); extern float \_\_sqrtf (float \_\_x) throw ();

/\* Return `sqrt(X\*X + Y\*Y)'. \*/

extern float hypotf (float \_\_x, float \_\_y) throw (); extern float \_\_hypotf (float \_\_x, float \_\_y) throw ();

/\* Return the cube root of X. \*/

extern float cbrtf (float \_\_x) throw (); extern float \_\_cbrtf (float \_\_x) throw ();

/\* Nearest integer, absolute value, and remainder functions. \*/

/\* Smallest integral value not less than X. \*/

extern float ceilf (float \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern float \_\_ceilf (float \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Absolute value of X. \*/

extern float fabsf (float \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern float \_\_fabsf (float \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Largest integer not greater than X. \*/

extern float floorf (float \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern float \_\_floorf (float \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Floating-point modulo remainder of X/Y. \*/

extern float fmodf (float \_\_x, float \_\_y) throw (); extern float \_\_fmodf (float \_\_x, float \_\_y) throw ();

/\* Return 0 if VALUE is finite or NaN, +1 if it

is +Infinity, -1 if it is -Infinity. \*/

extern int \_\_isinff (float \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is finite and not NaN. \*/

extern int \_\_finitef (float \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return 0 if VALUE is finite or NaN, +1 if it

is +Infinity, -1 if it is -Infinity. \*/

extern int isinff (float \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is finite and not NaN. \*/

extern int finitef (float \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return the remainder of X/Y. \*/

extern float dremf (float \_\_x, float \_\_y) throw (); extern float \_\_dremf (float \_\_x, float \_\_y) throw ();

/\* Return the fractional part of X after dividing out `ilogb (X)'. \*/

extern float significandf (float \_\_x) throw (); extern float \_\_significandf (float \_\_x) throw ();

/\* Return X with its signed changed to Y's. \*/

extern float copysignf (float \_\_x, float \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern float \_\_copysignf (float \_\_x, float \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return representation of NaN for double type. \*/

extern float nanf (\_\_const char \*\_\_tagb) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern float \_\_nanf (\_\_const char \*\_\_tagb) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is not a number. \*/

extern int \_\_isnanf (float \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is not a number. \*/

extern int isnanf (float \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Bessel functions. \*/

extern float j0f (float) throw (); extern float \_\_j0f (float) throw ();

extern float j1f (float) throw (); extern float \_\_j1f (float) throw ();

extern float jnf (int, float) throw (); extern float \_\_jnf (int, float) throw ();

extern float y0f (float) throw (); extern float \_\_y0f (float) throw ();

extern float y1f (float) throw (); extern float \_\_y1f (float) throw ();

extern float ynf (int, float) throw (); extern float \_\_ynf (int, float) throw ();

/\* Error and gamma functions. \*/

extern float erff (float) throw (); extern float \_\_erff (float) throw ();

extern float erfcf (float) throw (); extern float \_\_erfcf (float) throw ();

extern float lgammaf (float) throw (); extern float \_\_lgammaf (float) throw ();

/\* True gamma function. \*/

extern float tgammaf (float) throw (); extern float \_\_tgammaf (float) throw ();

/\* Obsolete alias for `lgamma'. \*/

extern float gammaf (float) throw (); extern float \_\_gammaf (float) throw ();

/\* Reentrant version of lgamma. This function uses the global variable

`signgam'. The reentrant version instead takes a pointer and stores

the value through it. \*/

extern float lgammaf\_r (float, int \*\_\_signgamp) throw (); extern float \_\_lgammaf\_r (float, int \*\_\_signgamp) throw ();

/\* Return the integer nearest X in the direction of the

prevailing rounding mode. \*/

extern float rintf (float \_\_x) throw (); extern float \_\_rintf (float \_\_x) throw ();

/\* Return X + epsilon if X < Y, X - epsilon if X > Y. \*/

extern float nextafterf (float \_\_x, float \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern float \_\_nextafterf (float \_\_x, float \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_));

extern float nexttowardf (float \_\_x, long double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern float \_\_nexttowardf (float \_\_x, long double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return the remainder of integer divison X / Y with infinite precision. \*/

extern float remainderf (float \_\_x, float \_\_y) throw (); extern float \_\_remainderf (float \_\_x, float \_\_y) throw ();

/\* Return X times (2 to the Nth power). \*/

extern float scalbnf (float \_\_x, int \_\_n) throw (); extern float \_\_scalbnf (float \_\_x, int \_\_n) throw ();

/\* Return the binary exponent of X, which must be nonzero. \*/

extern int ilogbf (float \_\_x) throw (); extern int \_\_ilogbf (float \_\_x) throw ();

/\* Return X times (2 to the Nth power). \*/

extern float scalblnf (float \_\_x, long int \_\_n) throw (); extern float \_\_scalblnf (float \_\_x, long int \_\_n) throw ();

/\* Round X to integral value in floating-point format using current

rounding direction, but do not raise inexact exception. \*/

extern float nearbyintf (float \_\_x) throw (); extern float \_\_nearbyintf (float \_\_x) throw ();

/\* Round X to nearest integral value, rounding halfway cases away from

zero. \*/

extern float roundf (float \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern float \_\_roundf (float \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Round X to the integral value in floating-point format nearest but

not larger in magnitude. \*/

extern float truncf (float \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern float \_\_truncf (float \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Compute remainder of X and Y and put in \*QUO a value with sign of x/y

and magnitude congruent `mod 2^n' to the magnitude of the integral

quotient x/y, with n >= 3. \*/

extern float remquof (float \_\_x, float \_\_y, int \*\_\_quo) throw (); extern float \_\_remquof (float \_\_x, float \_\_y, int \*\_\_quo) throw ();

/\* Conversion functions. \*/

/\* Round X to nearest integral value according to current rounding

direction. \*/

extern long int lrintf (float \_\_x) throw (); extern long int \_\_lrintf (float \_\_x) throw ();

extern long long int llrintf (float \_\_x) throw (); extern long long int \_\_llrintf (float \_\_x) throw ();

/\* Round X to nearest integral value, rounding halfway cases away from

zero. \*/

extern long int lroundf (float \_\_x) throw (); extern long int \_\_lroundf (float \_\_x) throw ();

extern long long int llroundf (float \_\_x) throw (); extern long long int \_\_llroundf (float \_\_x) throw ();

/\* Return positive difference between X and Y. \*/

extern float fdimf (float \_\_x, float \_\_y) throw (); extern float \_\_fdimf (float \_\_x, float \_\_y) throw ();

/\* Return maximum numeric value from X and Y. \*/

extern float fmaxf (float \_\_x, float \_\_y) throw (); extern float \_\_fmaxf (float \_\_x, float \_\_y) throw ();

/\* Return minimum numeric value from X and Y. \*/

extern float fminf (float \_\_x, float \_\_y) throw (); extern float \_\_fminf (float \_\_x, float \_\_y) throw ();

/\* Classify given number. \*/

extern int \_\_fpclassifyf (float \_\_value) throw ()

\_\_attribute\_\_ ((\_\_const\_\_));

/\* Test for negative number. \*/

extern int \_\_signbitf (float \_\_value) throw ()

\_\_attribute\_\_ ((\_\_const\_\_));

/\* Multiply-add function computed as a ternary operation. \*/

extern float fmaf (float \_\_x, float \_\_y, float \_\_z) throw (); extern float \_\_fmaf (float \_\_x, float \_\_y, float \_\_z) throw ();

/\* Return X times (2 to the Nth power). \*/

extern float scalbf (float \_\_x, float \_\_n) throw (); extern float \_\_scalbf (float \_\_x, float \_\_n) throw ();

#95 "/usr/include/math.h" 2 3 4

#131 "/usr/include/math.h" 3 4

/\* Include the file of declarations again, this time using `long double'

instead of `double' and appending l to each function name. \*/

#146 "/usr/include/math.h" 3 4

#1 "/usr/include/x86\_64-linux-gnu/bits/mathcalls.h" 1 3 4

/\* Prototype declarations for math functions; helper file for <math.h>.

Copyright (C) 1996-2002, 2003, 2006, 2011 Free Software Foundation, Inc.

This file is part of the GNU C Library.

The GNU C Library is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

The GNU C Library is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public

License along with the GNU C Library; if not, write to the Free

Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA

02111-1307 USA. \*/

/\* NOTE: Because of the special way this file is used by <math.h>, this

file must NOT be protected from multiple inclusion as header files

usually are.

This file provides prototype declarations for the math functions.

Most functions are declared using the macro:

\_\_MATHCALL (NAME,[\_r], (ARGS...));

This means there is a function `NAME' returning `double' and a function

`NAMEf' returning `float'. Each place `\_Mdouble\_' appears in the

prototype, that is actually `double' in the prototype for `NAME' and

`float' in the prototype for `NAMEf'. Reentrant variant functions are

called `NAME\_r' and `NAMEf\_r'.

Functions returning other types like `int' are declared using the macro:

\_\_MATHDECL (TYPE, NAME,[\_r], (ARGS...));

This is just like \_\_MATHCALL but for a function returning `TYPE'

instead of `\_Mdouble\_'. In all of these cases, there is still

both a `NAME' and a `NAMEf' that takes `float' arguments.

Note that there must be no whitespace before the argument passed for

NAME, to make token pasting work with -traditional. \*/

/\* Trigonometric functions. \*/

/\* Arc cosine of X. \*/

extern long double acosl (long double \_\_x) throw (); extern long double \_\_acosl (long double \_\_x) throw ();

/\* Arc sine of X. \*/

extern long double asinl (long double \_\_x) throw (); extern long double \_\_asinl (long double \_\_x) throw ();

/\* Arc tangent of X. \*/

extern long double atanl (long double \_\_x) throw (); extern long double \_\_atanl (long double \_\_x) throw ();

/\* Arc tangent of Y/X. \*/

extern long double atan2l (long double \_\_y, long double \_\_x) throw (); extern long double \_\_atan2l (long double \_\_y, long double \_\_x) throw ();

/\* Cosine of X. \*/

extern long double cosl (long double \_\_x) throw (); extern long double \_\_cosl (long double \_\_x) throw ();

/\* Sine of X. \*/

extern long double sinl (long double \_\_x) throw (); extern long double \_\_sinl (long double \_\_x) throw ();

/\* Tangent of X. \*/

extern long double tanl (long double \_\_x) throw (); extern long double \_\_tanl (long double \_\_x) throw ();

/\* Hyperbolic functions. \*/

/\* Hyperbolic cosine of X. \*/

extern long double coshl (long double \_\_x) throw (); extern long double \_\_coshl (long double \_\_x) throw ();

/\* Hyperbolic sine of X. \*/

extern long double sinhl (long double \_\_x) throw (); extern long double \_\_sinhl (long double \_\_x) throw ();

/\* Hyperbolic tangent of X. \*/

extern long double tanhl (long double \_\_x) throw (); extern long double \_\_tanhl (long double \_\_x) throw ();

/\* Cosine and sine of X. \*/

extern void sincosl (long double \_\_x, long double \*\_\_sinx, long double \*\_\_cosx) throw (); extern void \_\_sincosl (long double \_\_x, long double \*\_\_sinx, long double \*\_\_cosx) throw ();

/\* Hyperbolic arc cosine of X. \*/

extern long double acoshl (long double \_\_x) throw (); extern long double \_\_acoshl (long double \_\_x) throw ();

/\* Hyperbolic arc sine of X. \*/

extern long double asinhl (long double \_\_x) throw (); extern long double \_\_asinhl (long double \_\_x) throw ();

/\* Hyperbolic arc tangent of X. \*/

extern long double atanhl (long double \_\_x) throw (); extern long double \_\_atanhl (long double \_\_x) throw ();

/\* Exponential and logarithmic functions. \*/

/\* Exponential function of X. \*/

extern long double expl (long double \_\_x) throw (); extern long double \_\_expl (long double \_\_x) throw ();

/\* Break VALUE into a normalized fraction and an integral power of 2. \*/

extern long double frexpl (long double \_\_x, int \*\_\_exponent) throw (); extern long double \_\_frexpl (long double \_\_x, int \*\_\_exponent) throw ();

/\* X times (two to the EXP power). \*/

extern long double ldexpl (long double \_\_x, int \_\_exponent) throw (); extern long double \_\_ldexpl (long double \_\_x, int \_\_exponent) throw ();

/\* Natural logarithm of X. \*/

extern long double logl (long double \_\_x) throw (); extern long double \_\_logl (long double \_\_x) throw ();

/\* Base-ten logarithm of X. \*/

extern long double log10l (long double \_\_x) throw (); extern long double \_\_log10l (long double \_\_x) throw ();

/\* Break VALUE into integral and fractional parts. \*/

extern long double modfl (long double \_\_x, long double \*\_\_iptr) throw (); extern long double \_\_modfl (long double \_\_x, long double \*\_\_iptr) throw ()

\_\_attribute\_\_ ((\_\_nonnull\_\_ (2)));

/\* A function missing in all standards: compute exponent to base ten. \*/

extern long double exp10l (long double \_\_x) throw (); extern long double \_\_exp10l (long double \_\_x) throw ();

/\* Another name occasionally used. \*/

extern long double pow10l (long double \_\_x) throw (); extern long double \_\_pow10l (long double \_\_x) throw ();

/\* Return exp(X) - 1. \*/

extern long double expm1l (long double \_\_x) throw (); extern long double \_\_expm1l (long double \_\_x) throw ();

/\* Return log(1 + X). \*/

extern long double log1pl (long double \_\_x) throw (); extern long double \_\_log1pl (long double \_\_x) throw ();

/\* Return the base 2 signed integral exponent of X. \*/

extern long double logbl (long double \_\_x) throw (); extern long double \_\_logbl (long double \_\_x) throw ();

/\* Compute base-2 exponential of X. \*/

extern long double exp2l (long double \_\_x) throw (); extern long double \_\_exp2l (long double \_\_x) throw ();

/\* Compute base-2 logarithm of X. \*/

extern long double log2l (long double \_\_x) throw (); extern long double \_\_log2l (long double \_\_x) throw ();

/\* Power functions. \*/

/\* Return X to the Y power. \*/

extern long double powl (long double \_\_x, long double \_\_y) throw (); extern long double \_\_powl (long double \_\_x, long double \_\_y) throw ();

/\* Return the square root of X. \*/

extern long double sqrtl (long double \_\_x) throw (); extern long double \_\_sqrtl (long double \_\_x) throw ();

/\* Return `sqrt(X\*X + Y\*Y)'. \*/

extern long double hypotl (long double \_\_x, long double \_\_y) throw (); extern long double \_\_hypotl (long double \_\_x, long double \_\_y) throw ();

/\* Return the cube root of X. \*/

extern long double cbrtl (long double \_\_x) throw (); extern long double \_\_cbrtl (long double \_\_x) throw ();

/\* Nearest integer, absolute value, and remainder functions. \*/

/\* Smallest integral value not less than X. \*/

extern long double ceill (long double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern long double \_\_ceill (long double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Absolute value of X. \*/

extern long double fabsl (long double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern long double \_\_fabsl (long double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Largest integer not greater than X. \*/

extern long double floorl (long double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern long double \_\_floorl (long double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Floating-point modulo remainder of X/Y. \*/

extern long double fmodl (long double \_\_x, long double \_\_y) throw (); extern long double \_\_fmodl (long double \_\_x, long double \_\_y) throw ();

/\* Return 0 if VALUE is finite or NaN, +1 if it

is +Infinity, -1 if it is -Infinity. \*/

extern int \_\_isinfl (long double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is finite and not NaN. \*/

extern int \_\_finitel (long double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return 0 if VALUE is finite or NaN, +1 if it

is +Infinity, -1 if it is -Infinity. \*/

extern int isinfl (long double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is finite and not NaN. \*/

extern int finitel (long double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return the remainder of X/Y. \*/

extern long double dreml (long double \_\_x, long double \_\_y) throw (); extern long double \_\_dreml (long double \_\_x, long double \_\_y) throw ();

/\* Return the fractional part of X after dividing out `ilogb (X)'. \*/

extern long double significandl (long double \_\_x) throw (); extern long double \_\_significandl (long double \_\_x) throw ();

/\* Return X with its signed changed to Y's. \*/

extern long double copysignl (long double \_\_x, long double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern long double \_\_copysignl (long double \_\_x, long double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return representation of NaN for double type. \*/

extern long double nanl (\_\_const char \*\_\_tagb) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern long double \_\_nanl (\_\_const char \*\_\_tagb) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is not a number. \*/

extern int \_\_isnanl (long double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return nonzero if VALUE is not a number. \*/

extern int isnanl (long double \_\_value) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Bessel functions. \*/

extern long double j0l (long double) throw (); extern long double \_\_j0l (long double) throw ();

extern long double j1l (long double) throw (); extern long double \_\_j1l (long double) throw ();

extern long double jnl (int, long double) throw (); extern long double \_\_jnl (int, long double) throw ();

extern long double y0l (long double) throw (); extern long double \_\_y0l (long double) throw ();

extern long double y1l (long double) throw (); extern long double \_\_y1l (long double) throw ();

extern long double ynl (int, long double) throw (); extern long double \_\_ynl (int, long double) throw ();

/\* Error and gamma functions. \*/

extern long double erfl (long double) throw (); extern long double \_\_erfl (long double) throw ();

extern long double erfcl (long double) throw (); extern long double \_\_erfcl (long double) throw ();

extern long double lgammal (long double) throw (); extern long double \_\_lgammal (long double) throw ();

/\* True gamma function. \*/

extern long double tgammal (long double) throw (); extern long double \_\_tgammal (long double) throw ();

/\* Obsolete alias for `lgamma'. \*/

extern long double gammal (long double) throw (); extern long double \_\_gammal (long double) throw ();

/\* Reentrant version of lgamma. This function uses the global variable

`signgam'. The reentrant version instead takes a pointer and stores

the value through it. \*/

extern long double lgammal\_r (long double, int \*\_\_signgamp) throw (); extern long double \_\_lgammal\_r (long double, int \*\_\_signgamp) throw ();

/\* Return the integer nearest X in the direction of the

prevailing rounding mode. \*/

extern long double rintl (long double \_\_x) throw (); extern long double \_\_rintl (long double \_\_x) throw ();

/\* Return X + epsilon if X < Y, X - epsilon if X > Y. \*/

extern long double nextafterl (long double \_\_x, long double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern long double \_\_nextafterl (long double \_\_x, long double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_));

extern long double nexttowardl (long double \_\_x, long double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern long double \_\_nexttowardl (long double \_\_x, long double \_\_y) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Return the remainder of integer divison X / Y with infinite precision. \*/

extern long double remainderl (long double \_\_x, long double \_\_y) throw (); extern long double \_\_remainderl (long double \_\_x, long double \_\_y) throw ();

/\* Return X times (2 to the Nth power). \*/

extern long double scalbnl (long double \_\_x, int \_\_n) throw (); extern long double \_\_scalbnl (long double \_\_x, int \_\_n) throw ();

/\* Return the binary exponent of X, which must be nonzero. \*/

extern int ilogbl (long double \_\_x) throw (); extern int \_\_ilogbl (long double \_\_x) throw ();

/\* Return X times (2 to the Nth power). \*/

extern long double scalblnl (long double \_\_x, long int \_\_n) throw (); extern long double \_\_scalblnl (long double \_\_x, long int \_\_n) throw ();

/\* Round X to integral value in floating-point format using current

rounding direction, but do not raise inexact exception. \*/

extern long double nearbyintl (long double \_\_x) throw (); extern long double \_\_nearbyintl (long double \_\_x) throw ();

/\* Round X to nearest integral value, rounding halfway cases away from

zero. \*/

extern long double roundl (long double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern long double \_\_roundl (long double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Round X to the integral value in floating-point format nearest but

not larger in magnitude. \*/

extern long double truncl (long double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_)); extern long double \_\_truncl (long double \_\_x) throw () \_\_attribute\_\_ ((\_\_const\_\_));

/\* Compute remainder of X and Y and put in \*QUO a value with sign of x/y

and magnitude congruent `mod 2^n' to the magnitude of the integral

quotient x/y, with n >= 3. \*/

extern long double remquol (long double \_\_x, long double \_\_y, int \*\_\_quo) throw (); extern long double \_\_remquol (long double \_\_x, long double \_\_y, int \*\_\_quo) throw ();

/\* Conversion functions. \*/

/\* Round X to nearest integral value according to current rounding

direction. \*/

extern long int lrintl (long double \_\_x) throw (); extern long int \_\_lrintl (long double \_\_x) throw ();

extern long long int llrintl (long double \_\_x) throw (); extern long long int \_\_llrintl (long double \_\_x) throw ();

/\* Round X to nearest integral value, rounding halfway cases away from

zero. \*/

extern long int lroundl (long double \_\_x) throw (); extern long int \_\_lroundl (long double \_\_x) throw ();

extern long long int llroundl (long double \_\_x) throw (); extern long long int \_\_llroundl (long double \_\_x) throw ();

/\* Return positive difference between X and Y. \*/

extern long double fdiml (long double \_\_x, long double \_\_y) throw (); extern long double \_\_fdiml (long double \_\_x, long double \_\_y) throw ();

/\* Return maximum numeric value from X and Y. \*/

extern long double fmaxl (long double \_\_x, long double \_\_y) throw (); extern long double \_\_fmaxl (long double \_\_x, long double \_\_y) throw ();

/\* Return minimum numeric value from X and Y. \*/

extern long double fminl (long double \_\_x, long double \_\_y) throw (); extern long double \_\_fminl (long double \_\_x, long double \_\_y) throw ();

/\* Classify given number. \*/

extern int \_\_fpclassifyl (long double \_\_value) throw ()

\_\_attribute\_\_ ((\_\_const\_\_));

/\* Test for negative number. \*/

extern int \_\_signbitl (long double \_\_value) throw ()

\_\_attribute\_\_ ((\_\_const\_\_));

/\* Multiply-add function computed as a ternary operation. \*/

extern long double fmal (long double \_\_x, long double \_\_y, long double \_\_z) throw (); extern long double \_\_fmal (long double \_\_x, long double \_\_y, long double \_\_z) throw ();

/\* Return X times (2 to the Nth power). \*/

extern long double scalbl (long double \_\_x, long double \_\_n) throw (); extern long double \_\_scalbl (long double \_\_x, long double \_\_n) throw ();

#147 "/usr/include/math.h" 2 3 4

#161 "/usr/include/math.h" 3 4

/\* This variable is used by `gamma' and `lgamma'. \*/

extern int signgam;

/\* ISO C99 defines some generic macros which work on any data type. \*/

/\* Get the architecture specific values describing the floating-point

evaluation. The following symbols will get defined:

float\_t floating-point type at least as wide as `float' used

to evaluate `float' expressions

double\_t floating-point type at least as wide as `double' used

to evaluate `double' expressions

FLT\_EVAL\_METHOD

Defined to

0 if `float\_t' is `float' and `double\_t' is `double'

1 if `float\_t' and `double\_t' are `double'

2 if `float\_t' and `double\_t' are `long double'

else `float\_t' and `double\_t' are unspecified

INFINITY representation of the infinity value of type `float'

FP\_FAST\_FMA

FP\_FAST\_FMAF

FP\_FAST\_FMAL

If defined it indicates that the `fma' function

generally executes about as fast as a multiply and an add.

This macro is defined only iff the `fma' function is

implemented directly with a hardware multiply-add instructions.

FP\_ILOGB0 Expands to a value returned by `ilogb (0.0)'.

FP\_ILOGBNAN Expands to a value returned by `ilogb (NAN)'.

DECIMAL\_DIG Number of decimal digits supported by conversion between

decimal and all internal floating-point formats.

\*/

/\* All floating-point numbers can be put in one of these categories. \*/

enum

{

FP\_NAN,

FP\_INFINITE,

FP\_ZERO,

FP\_SUBNORMAL,

FP\_NORMAL

};

/\* Return number of classification appropriate for X. \*/

#229 "/usr/include/math.h" 3 4

/\* Return nonzero value if sign of X is negative. \*/

#241 "/usr/include/math.h" 3 4

/\* Return nonzero value if X is not +-Inf or NaN. \*/

#253 "/usr/include/math.h" 3 4

/\* Return nonzero value if X is neither zero, subnormal, Inf, nor NaN. \*/

/\* Return nonzero value if X is a NaN. We could use `fpclassify' but

we already have this functions `\_\_isnan' and it is faster. \*/

#269 "/usr/include/math.h" 3 4

/\* Return nonzero value if X is positive or negative infinity. \*/

#281 "/usr/include/math.h" 3 4

/\* Bitmasks for the math\_errhandling macro. \*/

/\* By default all functions support both errno and exception handling.

In gcc's fast math mode and if inline functions are defined this

might not be true. \*/

/\* Support for various different standard error handling behaviors. \*/

typedef enum

{

\_IEEE\_ = -1, /\* According to IEEE 754/IEEE 854. \*/

\_SVID\_, /\* According to System V, release 4. \*/

\_XOPEN\_, /\* Nowadays also Unix98. \*/

\_POSIX\_,

\_ISOC\_ /\* Actually this is ISO C99. \*/

} \_LIB\_VERSION\_TYPE;

/\* This variable can be changed at run-time to any of the values above to

affect floating point error handling behavior (it may also be necessary

to change the hardware FPU exception settings). \*/

extern \_LIB\_VERSION\_TYPE \_LIB\_VERSION;

/\* In SVID error handling, `matherr' is called with this description

of the exceptional condition.

We have a problem when using C++ since `exception' is a reserved

name in C++. \*/

struct \_\_exception

{

int type;

char \*name;

double arg1;

double arg2;

double retval;

};

extern int matherr (struct \_\_exception \*\_\_exc) throw ();

/\* Types of exceptions in the `type' field. \*/

/\* SVID mode specifies returning this large value instead of infinity. \*/

#360 "/usr/include/math.h" 3 4

/\* Some useful constants. \*/

#377 "/usr/include/math.h" 3 4

/\* The above constants are not adequate for computation using `long double's.

Therefore we provide as an extension constants with similar names as a

GNU extension. Provide enough digits for the 128-bit IEEE quad. \*/

#397 "/usr/include/math.h" 3 4

/\* When compiling in strict ISO C compatible mode we must not use the

inline functions since they, among other things, do not set the

`errno' variable correctly. \*/

/\* ISO C99 defines some macros to compare number while taking care for

unordered numbers. Many FPUs provide special instructions to support

these operations. Generic support in GCC for these as builtins went

in before 3.0.0, but not all cpus added their patterns. We define

versions that use the builtins here, and <bits/mathinline.h> will

undef/redefine as appropriate for the specific GCC version in use. \*/

#419 "/usr/include/math.h" 3 4

/\* Get machine-dependent inline versions (if there are any). \*/

/\* Define special entry points to use when the compiler got told to

only expect finite results. \*/

/\* If we've still got undefined comparison macros, provide defaults. \*/

/\* Return nonzero value if X is greater than Y. \*/

/\* Return nonzero value if X is greater than or equal to Y. \*/

/\* Return nonzero value if X is less than Y. \*/

/\* Return nonzero value if X is less than or equal to Y. \*/

/\* Return nonzero value if either X is less than Y or Y is less than X. \*/

/\* Return nonzero value if arguments are unordered. \*/

#483 "/usr/include/math.h" 3 4

}

#46 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/cmath" 2 3

}

#46 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/cmath"

// Get rid of those macros defined in <math.h> in lieu of real functions.

#76 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/cmath" 3

namespace std \_\_attribute\_\_ ((\_\_visibility\_\_ ("default")))

{

inline double

abs(double \_\_x)

{ return \_\_builtin\_fabs(\_\_x); }

inline float

abs(float \_\_x)

{ return \_\_builtin\_fabsf(\_\_x); }

inline long double

abs(long double \_\_x)

{ return \_\_builtin\_fabsl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

abs(\_Tp \_\_x)

{ return \_\_builtin\_fabs(\_\_x); }

using ::acos;

inline float

acos(float \_\_x)

{ return \_\_builtin\_acosf(\_\_x); }

inline long double

acos(long double \_\_x)

{ return \_\_builtin\_acosl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

acos(\_Tp \_\_x)

{ return \_\_builtin\_acos(\_\_x); }

using ::asin;

inline float

asin(float \_\_x)

{ return \_\_builtin\_asinf(\_\_x); }

inline long double

asin(long double \_\_x)

{ return \_\_builtin\_asinl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

asin(\_Tp \_\_x)

{ return \_\_builtin\_asin(\_\_x); }

using ::atan;

inline float

atan(float \_\_x)

{ return \_\_builtin\_atanf(\_\_x); }

inline long double

atan(long double \_\_x)

{ return \_\_builtin\_atanl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

atan(\_Tp \_\_x)

{ return \_\_builtin\_atan(\_\_x); }

using ::atan2;

inline float

atan2(float \_\_y, float \_\_x)

{ return \_\_builtin\_atan2f(\_\_y, \_\_x); }

inline long double

atan2(long double \_\_y, long double \_\_x)

{ return \_\_builtin\_atan2l(\_\_y, \_\_x); }

template<typename \_Tp, typename \_Up>

inline

typename \_\_gnu\_cxx::\_\_promote\_2<\_Tp, \_Up>::\_\_type

atan2(\_Tp \_\_y, \_Up \_\_x)

{

typedef typename \_\_gnu\_cxx::\_\_promote\_2<\_Tp, \_Up>::\_\_type \_\_type;

return atan2(\_\_type(\_\_y), \_\_type(\_\_x));

}

using ::ceil;

inline float

ceil(float \_\_x)

{ return \_\_builtin\_ceilf(\_\_x); }

inline long double

ceil(long double \_\_x)

{ return \_\_builtin\_ceill(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

ceil(\_Tp \_\_x)

{ return \_\_builtin\_ceil(\_\_x); }

using ::cos;

inline float

cos(float \_\_x)

{ return \_\_builtin\_cosf(\_\_x); }

inline long double

cos(long double \_\_x)

{ return \_\_builtin\_cosl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

cos(\_Tp \_\_x)

{ return \_\_builtin\_cos(\_\_x); }

using ::cosh;

inline float

cosh(float \_\_x)

{ return \_\_builtin\_coshf(\_\_x); }

inline long double

cosh(long double \_\_x)

{ return \_\_builtin\_coshl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

cosh(\_Tp \_\_x)

{ return \_\_builtin\_cosh(\_\_x); }

using ::exp;

inline float

exp(float \_\_x)

{ return \_\_builtin\_expf(\_\_x); }

inline long double

exp(long double \_\_x)

{ return \_\_builtin\_expl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

exp(\_Tp \_\_x)

{ return \_\_builtin\_exp(\_\_x); }

using ::fabs;

inline float

fabs(float \_\_x)

{ return \_\_builtin\_fabsf(\_\_x); }

inline long double

fabs(long double \_\_x)

{ return \_\_builtin\_fabsl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

fabs(\_Tp \_\_x)

{ return \_\_builtin\_fabs(\_\_x); }

using ::floor;

inline float

floor(float \_\_x)

{ return \_\_builtin\_floorf(\_\_x); }

inline long double

floor(long double \_\_x)

{ return \_\_builtin\_floorl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

floor(\_Tp \_\_x)

{ return \_\_builtin\_floor(\_\_x); }

using ::fmod;

inline float

fmod(float \_\_x, float \_\_y)

{ return \_\_builtin\_fmodf(\_\_x, \_\_y); }

inline long double

fmod(long double \_\_x, long double \_\_y)

{ return \_\_builtin\_fmodl(\_\_x, \_\_y); }

using ::frexp;

inline float

frexp(float \_\_x, int\* \_\_exp)

{ return \_\_builtin\_frexpf(\_\_x, \_\_exp); }

inline long double

frexp(long double \_\_x, int\* \_\_exp)

{ return \_\_builtin\_frexpl(\_\_x, \_\_exp); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

frexp(\_Tp \_\_x, int\* \_\_exp)

{ return \_\_builtin\_frexp(\_\_x, \_\_exp); }

using ::ldexp;

inline float

ldexp(float \_\_x, int \_\_exp)

{ return \_\_builtin\_ldexpf(\_\_x, \_\_exp); }

inline long double

ldexp(long double \_\_x, int \_\_exp)

{ return \_\_builtin\_ldexpl(\_\_x, \_\_exp); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

ldexp(\_Tp \_\_x, int \_\_exp)

{ return \_\_builtin\_ldexp(\_\_x, \_\_exp); }

using ::log;

inline float

log(float \_\_x)

{ return \_\_builtin\_logf(\_\_x); }

inline long double

log(long double \_\_x)

{ return \_\_builtin\_logl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

log(\_Tp \_\_x)

{ return \_\_builtin\_log(\_\_x); }

using ::log10;

inline float

log10(float \_\_x)

{ return \_\_builtin\_log10f(\_\_x); }

inline long double

log10(long double \_\_x)

{ return \_\_builtin\_log10l(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

log10(\_Tp \_\_x)

{ return \_\_builtin\_log10(\_\_x); }

using ::modf;

inline float

modf(float \_\_x, float\* \_\_iptr)

{ return \_\_builtin\_modff(\_\_x, \_\_iptr); }

inline long double

modf(long double \_\_x, long double\* \_\_iptr)

{ return \_\_builtin\_modfl(\_\_x, \_\_iptr); }

using ::pow;

inline float

pow(float \_\_x, float \_\_y)

{ return \_\_builtin\_powf(\_\_x, \_\_y); }

inline long double

pow(long double \_\_x, long double \_\_y)

{ return \_\_builtin\_powl(\_\_x, \_\_y); }

// \_GLIBCXX\_RESOLVE\_LIB\_DEFECTS

// DR 550. What should the return type of pow(float,int) be?

inline double

pow(double \_\_x, int \_\_i)

{ return \_\_builtin\_powi(\_\_x, \_\_i); }

inline float

pow(float \_\_x, int \_\_n)

{ return \_\_builtin\_powif(\_\_x, \_\_n); }

inline long double

pow(long double \_\_x, int \_\_n)

{ return \_\_builtin\_powil(\_\_x, \_\_n); }

template<typename \_Tp, typename \_Up>

inline

typename \_\_gnu\_cxx::\_\_promote\_2<\_Tp, \_Up>::\_\_type

pow(\_Tp \_\_x, \_Up \_\_y)

{

typedef typename \_\_gnu\_cxx::\_\_promote\_2<\_Tp, \_Up>::\_\_type \_\_type;

return pow(\_\_type(\_\_x), \_\_type(\_\_y));

}

using ::sin;

inline float

sin(float \_\_x)

{ return \_\_builtin\_sinf(\_\_x); }

inline long double

sin(long double \_\_x)

{ return \_\_builtin\_sinl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

sin(\_Tp \_\_x)

{ return \_\_builtin\_sin(\_\_x); }

using ::sinh;

inline float

sinh(float \_\_x)

{ return \_\_builtin\_sinhf(\_\_x); }

inline long double

sinh(long double \_\_x)

{ return \_\_builtin\_sinhl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

sinh(\_Tp \_\_x)

{ return \_\_builtin\_sinh(\_\_x); }

using ::sqrt;

inline float

sqrt(float \_\_x)

{ return \_\_builtin\_sqrtf(\_\_x); }

inline long double

sqrt(long double \_\_x)

{ return \_\_builtin\_sqrtl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

sqrt(\_Tp \_\_x)

{ return \_\_builtin\_sqrt(\_\_x); }

using ::tan;

inline float

tan(float \_\_x)

{ return \_\_builtin\_tanf(\_\_x); }

inline long double

tan(long double \_\_x)

{ return \_\_builtin\_tanl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

tan(\_Tp \_\_x)

{ return \_\_builtin\_tan(\_\_x); }

using ::tanh;

inline float

tanh(float \_\_x)

{ return \_\_builtin\_tanhf(\_\_x); }

inline long double

tanh(long double \_\_x)

{ return \_\_builtin\_tanhl(\_\_x); }

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_integer<\_Tp>::\_\_value,

double>::\_\_type

tanh(\_Tp \_\_x)

{ return \_\_builtin\_tanh(\_\_x); }

} // namespace

// These are possible macros imported from C99-land.

#480 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/cmath" 3

namespace std \_\_attribute\_\_ ((\_\_visibility\_\_ ("default")))

{

#730 "/opt/Xilinx/Vivado\_HLS/2017.2/lnx64/tools/gcc/lib/gcc/x86\_64-unknown-linux-gnu/4.6.3/../../../../include/c++/4.6.3/cmath" 3

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

fpclassify(\_Tp \_\_f)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_fpclassify(FP\_NAN, FP\_INFINITE, FP\_NORMAL,

FP\_SUBNORMAL, FP\_ZERO, \_\_type(\_\_f));

}

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

isfinite(\_Tp \_\_f)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_isfinite(\_\_type(\_\_f));

}

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

isinf(\_Tp \_\_f)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_isinf(\_\_type(\_\_f));

}

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

isnan(\_Tp \_\_f)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_isnan(\_\_type(\_\_f));

}

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

isnormal(\_Tp \_\_f)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_isnormal(\_\_type(\_\_f));

}

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

signbit(\_Tp \_\_f)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_signbit(\_\_type(\_\_f));

}

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

isgreater(\_Tp \_\_f1, \_Tp \_\_f2)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_isgreater(\_\_type(\_\_f1), \_\_type(\_\_f2));

}

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

isgreaterequal(\_Tp \_\_f1, \_Tp \_\_f2)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_isgreaterequal(\_\_type(\_\_f1), \_\_type(\_\_f2));

}

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

isless(\_Tp \_\_f1, \_Tp \_\_f2)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_isless(\_\_type(\_\_f1), \_\_type(\_\_f2));

}

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

islessequal(\_Tp \_\_f1, \_Tp \_\_f2)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_islessequal(\_\_type(\_\_f1), \_\_type(\_\_f2));

}

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

islessgreater(\_Tp \_\_f1, \_Tp \_\_f2)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_islessgreater(\_\_type(\_\_f1), \_\_type(\_\_f2));

}

template<typename \_Tp>

inline typename \_\_gnu\_cxx::\_\_enable\_if<\_\_is\_arithmetic<\_Tp>::\_\_value,

int>::\_\_type

isunordered(\_Tp \_\_f1, \_Tp \_\_f2)

{

typedef typename \_\_gnu\_cxx::\_\_promote<\_Tp>::\_\_type \_\_type;

return \_\_builtin\_isunordered(\_\_type(\_\_f1), \_\_type(\_\_f2));

}

} // namespace

#70 "../../../../../../home/drsatya/Desktop/lab1/matrixmul.h" 2

using namespace std;

// Uncomment this line to compare TB vs HW C-model and/or RTL

//#define HW\_COSIM

typedef char mat\_a\_t;

typedef char mat\_b\_t;

typedef short result\_t;

// Prototype of top level function for C-synthesis

void matrixmul(

mat\_a\_t a[3][3],

mat\_b\_t b[3][3],

result\_t res[3][3]);

#3 "../../../../../../home/drsatya/Desktop/lab1/mat\_pipeline.cpp" 2

void matrixmul(

mat\_a\_t a[3][3],

mat\_b\_t b[3][3],

result\_t res[3][3])

{\_ssdm\_SpecArrayDimSize(a,3);\_ssdm\_SpecArrayDimSize(res,3);\_ssdm\_SpecArrayDimSize(b,3);

// Iterate over the rows of the A matrix

Row: for(int i = 0; i < 3; i++) {

// Iterate over the columns of the B matrix

Col: for(int j = 0; j < 3; j++) {

// Do the inner product of a row of A and col of B

res[i][j] = 0;

Product: for(int k = 0; k < 3; k++) {

\_ssdm\_op\_SpecPipeline(1, 1, 1, 0, "");

res[i][j] += a[i][k] \* b[k][j];

}

}

}

}