

Linux Standard Base Core Specification for X86-64

Linux Standard Base Core Specification for X86-64

LSB Core - X86-64 5.0

Copyright © 2015 Linux Foundation

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Portions of the text may be copyrighted by the following parties:

- The Regents of the University of California
- Free Software Foundation
- Ian F. Darwin
- Paul Vixie
- BSDI (now Wind River)
- Jean-loup Gailly and Mark Adler
- Massachusetts Institute of Technology
- Apple Inc.
- Easy Software Products
- artofcode LLC
- Till Kamppeter
- Manfred Wassman
- Python Software Foundation

These excerpts are being used in accordance with their respective licenses.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

UNIX is a registered trademark of The Open Group.

LSB is a trademark of the Linux Foundation in the United States and other countries.

AMD is a trademark of Advanced Micro Devices, Inc.

Intel and Itanium are registered trademarks and Intel386 is a trademark of Intel Corporation.

PowerPC is a registered trademark and PowerPC Architecture is a trademark of the IBM Corporation.

S/390 is a registered trademark of the IBM Corporation.

OpenGL is a registered trademark of Silicon Graphics, Inc.

PAM documentation is Copyright (C) Andrew G. Morgan 1996-9. All rights reserved. Used under the following conditions:

1. Redistributions of source code must retain the above copyright notice, and the entire permission notice in its entirety, including the disclaimer of warranties.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

Contents

<u>I Introductory Elements</u>	
<u>1 Scope</u>	
<u>1.1 General</u>	
<u>1.2 Module Specific Scope</u>	
<u>2 References</u>	
<u>2.1 Normative References</u>	
<u>2.2 Informative References/Bibliography</u>	
<u>3 Requirements</u>	
<u>3.1 Relevant Libraries</u>	
<u>3.2 LSB Implementation Conformance</u>	
<u>3.3 LSB Application Conformance</u>	
<u>4 Terms and Definitions</u>	
<u>5 Documentation Conventions</u>	
<u>II Executable and Linking Format (ELF)</u>	
<u>6 Introduction</u>	
<u>7 Low Level System Information</u>	
<u>7.1 Machine Interface</u>	
<u>7.2 Function Calling Sequence</u>	
<u>7.3 Operating System Interface</u>	
<u>7.4 Process Initialization</u>	
<u>7.5 Coding Examples</u>	
<u>7.6 C Stack Frame</u>	
<u>7.7 Debug Information</u>	
<u>8 Object Format</u>	
<u>8.1 Introduction</u>	
<u>8.2 ELF Header</u>	
<u>8.3 Sections</u>	
<u>8.4 Symbol Table</u>	
<u>8.5 Relocation</u>	
<u>9 Program Loading and Dynamic Linking</u>	
<u>9.1 Introduction</u>	
<u>9.2 Program Header</u>	
<u>9.3 Program Loading</u>	
<u>9.4 Dynamic Linking</u>	
<u>III Base Libraries</u>	
<u>10 Libraries</u>	
<u>10.1 Program Interpreter/Dynamic Linker</u>	
<u>10.2 Interfaces for libc</u>	
<u>10.3 Data Definitions for libc</u>	
<u>10.4 Interface Definitions for libc</u>	
<u>10.5 Interfaces for libm</u>	
<u>10.6 Data Definitions for libm</u>	
<u>10.7 Interface Definitions for libm</u>	
<u>10.8 Interfaces for libpthread</u>	
<u>10.9 Data Definitions for libpthread</u>	
<u>10.10 Interfaces for libgcc_s</u>	
<u>10.11 Data Definitions for libgcc_s</u>	
<u>10.12 Interface Definitions for libgcc_s</u>	
<u>10.13 Interfaces for libdl</u>	
<u>10.14 Data Definitions for libdl</u>	
<u>10.15 Interfaces for libcrypt</u>	
<u>10.16 Data Definitions for libcrypt</u>	
<u>IV Utility Libraries</u>	
<u>11 Libraries</u>	

- [11.1 Interfaces for libz.....](#)
- [11.2 Data Definitions for libz.....](#)
- [11.3 Interfaces for libncurses.....](#)
- [11.4 Data Definitions for libncurses.....](#)
- [11.5 Interfaces for libncursesw.....](#)
- [11.6 Data Definitions for libncursesw.....](#)
- [11.7 Interfaces for libutil.....](#)
- [V Base Libraries.....](#)**
 - [12 Libraries.....](#)
 - [12.1 Interfaces for libstdcxx.....](#)
 - [12.2 Interface Definitions for libstdcxx.....](#)
- [VI Package Format and Installation.....](#)**
 - [13 Software Installation.....](#)
 - [13.1 Package Dependencies.....](#)
 - [13.2 Package Architecture Considerations.....](#)
- [A Alphabetical Listing of Interfaces by Library.....](#)**
 - [A.1 libc.....](#)
 - [A.2 libcrypt.....](#)
 - [A.3 libdl.....](#)
 - [A.4 libgcc_s.....](#)
 - [A.5 libm.....](#)
 - [A.6 libpthread.....](#)
 - [A.7 librt.....](#)
 - [A.8 libutil.....](#)
- [B GNU Free Documentation License \(Informative\).....](#)**
 - [B.1 PREAMBLE.....](#)
 - [B.2 APPLICABILITY AND DEFINITIONS.....](#)
 - [B.3 VERBATIM COPYING.....](#)
 - [B.4 COPYING IN QUANTITY.....](#)
 - [B.5 MODIFICATIONS.....](#)
 - [B.6 COMBINING DOCUMENTS.....](#)
 - [B.7 COLLECTIONS OF DOCUMENTS.....](#)
 - [B.8 AGGREGATION WITH INDEPENDENT WORKS.....](#)
 - [B.9 TRANSLATION.....](#)
 - [B.10 TERMINATION.....](#)
 - [B.11 FUTURE REVISIONS OF THIS LICENSE.....](#)
 - [B.12 How to use this License for your documents.....](#)

List of Tables

2-1 Normative References
2-2 Other References
3-1 Standard Library Names
7-1 Non Conforming Instructions
8-1 ELF Special Sections
8-2 Additional Special Sections
10-1 libc Definition
10-2 libc - RPC Function Interfaces
10-3 libc - RPC Deprecated Function Interfaces
10-4 libc - System Calls Function Interfaces
10-5 libc - System Calls Deprecated Function Interfaces
10-6 libc - Standard I/O Function Interfaces
10-7 libc - Standard I/O Deprecated Function Interfaces
10-8 libc - Standard I/O Data Interfaces
10-9 libc - Signal Handling Function Interfaces
10-10 libc - Signal Handling Deprecated Function Interfaces
10-11 libc - Signal Handling Data Interfaces
10-12 libc - Localization Functions Function Interfaces
10-13 libc - Localization Functions Data Interfaces
10-14 libc - Posix Spawn Option Function Interfaces
10-15 libc - Posix Advisory Option Function Interfaces
10-16 libc - Socket Interface Function Interfaces
10-17 libc - Socket Interface Data Interfaces
10-18 libc - Wide Characters Function Interfaces
10-19 libc - String Functions Function Interfaces
10-20 libc - String Functions Deprecated Function Interfaces
10-21 libc - IPC Functions Function Interfaces
10-22 libc - Regular Expressions Function Interfaces
10-23 libc - Character Type Functions Function Interfaces
10-24 libc - Time Manipulation Function Interfaces
10-25 libc - Time Manipulation Data Interfaces
10-26 libc - Terminal Interface Functions Function Interfaces
10-27 libc - System Database Interface Function Interfaces
10-28 libc - System Database Interface Deprecated Function Interfaces
10-29 libc - Language Support Function Interfaces
10-30 libc - Large File Support Function Interfaces
10-31 libc - Large File Support Deprecated Function Interfaces
10-32 libc - Standard Library Function Interfaces
10-33 libc - Standard Library Deprecated Function Interfaces
10-34 libc - Standard Library Data Interfaces
10-35 libc - GNU Extensions for libc Function Interfaces
10-36 libm Definition
10-37 libm - Math Function Interfaces
10-38 libm - Math Deprecated Function Interfaces
10-39 libm - Math Data Interfaces
10-40 libpthread Definition
10-41 libpthread - Realtime Threads Function Interfaces
10-42 libpthread - Advanced Realtime Threads Function Interfaces
10-43 libpthread - Posix Threads Function Interfaces
10-44 libpthread - Posix Threads Deprecated Function Interfaces
10-45 libpthread - Thread aware versions of libc interfaces Function Interfaces
10-46 libpthread - GNU Extensions for libpthread Function Interfaces
10-47 libpthread - System Calls Function Interfaces
10-48 libpthread - Standard I/O Function Interfaces
10-49 libpthread - Signal Handling Function Interfaces

[10-50 libpthread - Standard Library Function Interfaces](#).....

[10-51 libpthread - Socket Interface Function Interfaces](#).....

[10-52 libpthread - Terminal Interface Functions Function Interfaces](#).....

[10-53 libgcc s Definition](#).....

[10-54 libgcc s - Unwind Library Function Interfaces](#).....

[10-55 libdl Definition](#).....

[10-56 libdl - Dynamic Loader Function Interfaces](#).....

[10-57 libcrypt Definition](#).....

[10-58 libcrypt - Encryption Function Interfaces](#).....

[11-1 libz Definition](#).....

[11-2 libncurses Definition](#).....

[11-3 libncursesw Definition](#).....

[11-4 libutil Definition](#).....

[11-5 libutil - Utility Functions Function Interfaces](#).....

[12-1 libstdcxx Definition](#).....

[12-2 libstdcxx - C++ Runtime Support Function Interfaces](#).....

[12-3 typeid for type info](#).....

[12-4 typeid for cxxabiv1:: enum type info](#).....

[12-5 typeid for cxxabiv1:: array type info](#).....

[12-6 Primary vtable for cxxabiv1:: class type info](#).....

[12-7 typeid for cxxabiv1:: class type info](#).....

[12-8 libstdcxx - Class cxxabiv1:: class type info Function Interfaces](#).....

[12-9 typeid for cxxabiv1:: pbase type info](#).....

[12-10 typeid for cxxabiv1:: pointer type info](#).....

[12-11 typeid for cxxabiv1:: function type info](#).....

[12-12 Primary vtable for cxxabiv1:: si class type info](#).....

[12-13 typeid for cxxabiv1:: si class type info](#).....

[12-14 libstdcxx - Class cxxabiv1:: si class type info Function Interfaces](#).....

[12-15 Primary vtable for cxxabiv1:: vmi class type info](#).....

[12-16 typeid for cxxabiv1:: vmi class type info](#).....

[12-17 libstdcxx - Class cxxabiv1:: vmi class type info Function Interfaces](#).....

[12-18 typeid for cxxabiv1:: fundamental type info](#).....

[12-19 typeid for cxxabiv1:: pointer to member type info](#).....

[12-20 libstdcxx - Class gnu cxx:: pool alloc base Function Interfaces](#).....

[12-21 Primary vtable for gnu cxx::stdio_sync filebuf<char, char traits<char>>](#).....

[12-22 Primary vtable for gnu cxx::stdio_sync filebuf<wchar t, char traits<wchar t>](#)
[>](#).....

[12-23 typeid for exception](#).....

[12-24 typeid for bad typeid](#).....

[12-25 typeid for logic error](#).....

[12-26 typeid for range error](#).....

[12-27 typeid for domain error](#).....

[12-28 typeid for length error](#).....

[12-29 typeid for out of range](#).....

[12-30 typeid for bad exception](#).....

[12-31 typeid for runtime error](#).....

[12-32 typeid for overflow error](#).....

[12-33 typeid for underflow error](#).....

[12-34 typeid for invalid argument](#).....

[12-35 typeid for bad cast](#).....

[12-36 typeid for bad alloc](#).....

[12-37 typeid for ctype base](#).....

[12-38 libstdcxx - Class ctype<char> Function Interfaces](#).....

[12-39 typeid for ctype<wchar t>](#).....

[12-40 libstdcxx - Class ctype<wchar t> Function Interfaces](#).....

[12-41 typeid for ctype byname<char>](#).....

[12-42 libstdcxx - Class ctype byname<char> Function Interfaces](#).....

[12-43 typeid for ctype byname<wchar t>](#).....

[12-44 libstdcxx - Class ctype_byname<wchar_t> Function Interfaces](#).....

[12-45 libstdcxx - Class basic_string<char, char_traits<char>, allocator<char>> Function Interfaces](#).....

[12-46 libstdcxx - Class basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> Function Interfaces](#).....

[12-47 Primary vtable for basic_stringstream<char, char_traits<char>, allocator<char>>](#).....

[12-48 Secondary vtable for basic_stringstream<char, char_traits<char>, allocator<char>>](#).....

[12-49 Secondary vtable for basic_stringstream<char, char_traits<char>, allocator<char>>](#).....

[12-50 VTT for basic_stringstream<char, char_traits<char>, allocator<char>>](#).....

[12-51 libstdcxx - Class basic_stringstream<char, char_traits<char>, allocator<char>> Function Interfaces](#).....

[12-52 Primary vtable for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>](#).....

[12-53 Secondary vtable for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>](#).....

[12-54 Secondary vtable for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>](#).....

[12-55 VTT for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>](#).....

[12-56 libstdcxx - Class basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> Function Interfaces](#).....

[12-57 Primary vtable for basic_istringstream<char, char_traits<char>, allocator<char>>](#).....

[12-58 Secondary vtable for basic_istringstream<char, char_traits<char>, allocator<char>>](#).....

[12-59 VTT for basic_istringstream<char, char_traits<char>, allocator<char>>](#).....

[12-60 libstdcxx - Class basic_istringstream<char, char_traits<char>, allocator<char>> Function Interfaces](#).....

[12-61 Primary vtable for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>](#).....

[12-62 Secondary vtable for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>](#).....

[12-63 VTT for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>](#).....

[12-64 libstdcxx - Class basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> Function Interfaces](#).....

[12-65 Primary vtable for basic_ostringstream<char, char_traits<char>, allocator<char>>](#).....

[12-66 Secondary vtable for basic_ostringstream<char, char_traits<char>, allocator<char>>](#).....

[12-67 VTT for basic_ostringstream<char, char_traits<char>, allocator<char>>](#).....

[12-68 libstdcxx - Class basic_ostringstream<char, char_traits<char>, allocator<char>> Function Interfaces](#).....

[12-69 Primary vtable for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>](#).....

[12-70 Secondary vtable for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>](#).....

[12-71 VTT for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>](#).....

[12-72 libstdcxx - Class basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> Function Interfaces](#).....

[12-73 Primary vtable for basic_stringbuf<char, char_traits<char>, allocator<char>>](#).....

[12-74 typeid for basic_stringbuf<char, char_traits<char>, allocator<char>>](#).....

[12-75 libstdcxx - Class basic_stringbuf<char, char_traits<char>, allocator<char>> Function Interfaces](#).....

[12-76 Primary vtable for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>](#).....

[12-77 typeid for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>.....](#)

[12-78 libstdcxx - Class basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> Function Interfaces.....](#)

[12-79 Primary vtable for basic_istream<char, char_traits<char>>.....](#)

[12-80 Secondary vtable for basic_istream<char, char_traits<char>>.....](#)

[12-81 Secondary vtable for basic_istream<char, char_traits<char>>.....](#)

[12-82 VTT for basic_istream<char, char_traits<char>>.....](#)

[12-83 libstdcxx - Class basic_istream<char, char_traits<char>> Function Interfaces.....](#)

[12-84 Primary vtable for basic_istream<wchar_t, char_traits<wchar_t>>.....](#)

[12-85 Secondary vtable for basic_istream<wchar_t, char_traits<wchar_t>>.....](#)

[12-86 Secondary vtable for basic_istream<wchar_t, char_traits<wchar_t>>.....](#)

[12-87 VTT for basic_istream<wchar_t, char_traits<wchar_t>>.....](#)

[12-88 libstdcxx - Class basic_istream<wchar_t, char_traits<wchar_t>> Function Interfaces.....](#)

[12-89 Primary vtable for basic_istream<char, char_traits<char>>.....](#)

[12-90 Secondary vtable for basic_istream<char, char_traits<char>>.....](#)

[12-91 VTT for basic_istream<char, char_traits<char>>.....](#)

[12-92 libstdcxx - Class basic_istream<char, char_traits<char>> Function Interfaces.....](#)

[12-93 Primary vtable for basic_istream<wchar_t, char_traits<wchar_t>>.....](#)

[12-94 Secondary vtable for basic_istream<wchar_t, char_traits<wchar_t>>.....](#)

[12-95 VTT for basic_istream<wchar_t, char_traits<wchar_t>>.....](#)

[12-96 libstdcxx - Class basic_istream<wchar_t, char_traits<wchar_t>> Function Interfaces.....](#)

[12-97 Primary vtable for basic_ostream<char, char_traits<char>>.....](#)

[12-98 Secondary vtable for basic_ostream<char, char_traits<char>>.....](#)

[12-99 VTT for basic_ostream<char, char_traits<char>>.....](#)

[12-100 libstdcxx - Class basic_ostream<char, char_traits<char>> Function Interfaces.....](#)

[12-101 Primary vtable for basic_ostream<wchar_t, char_traits<wchar_t>>.....](#)

[12-102 Secondary vtable for basic_ostream<wchar_t, char_traits<wchar_t>>.....](#)

[12-103 VTT for basic_ostream<wchar_t, char_traits<wchar_t>>.....](#)

[12-104 libstdcxx - Class basic_ostream<wchar_t, char_traits<wchar_t>> Function Interfaces.....](#)

[12-105 Primary vtable for basicfstream<char, char_traits<char>>.....](#)

[12-106 Secondary vtable for basicfstream<char, char_traits<char>>.....](#)

[12-107 Secondary vtable for basicfstream<char, char_traits<char>>.....](#)

[12-108 VTT for basicfstream<char, char_traits<char>>.....](#)

[12-109 libstdcxx - Class basicfstream<char, char_traits<char>> Function Interfaces.....](#)

[12-110 Primary vtable for basicfstream<wchar_t, char_traits<wchar_t>>.....](#)

[12-111 Secondary vtable for basicfstream<wchar_t, char_traits<wchar_t>>.....](#)

[12-112 Secondary vtable for basicfstream<wchar_t, char_traits<wchar_t>>.....](#)

[12-113 VTT for basicfstream<wchar_t, char_traits<wchar_t>>.....](#)

[12-114 libstdcxx - Class basicfstream<wchar_t, char_traits<wchar_t>> Function Interfaces.....](#)

[12-115 Primary vtable for basicifstream<char, char_traits<char>>.....](#)

[12-116 Secondary vtable for basicifstream<char, char_traits<char>>.....](#)

[12-117 VTT for basicifstream<char, char_traits<char>>.....](#)

[12-118 libstdcxx - Class basicifstream<char, char_traits<char>> Function Interfaces.....](#)

[12-119 Primary vtable for basicifstream<wchar_t, char_traits<wchar_t>>.....](#)

[12-120 Secondary vtable for basicifstream<wchar_t, char_traits<wchar_t>>.....](#)

[12-121 VTT for basicifstream<wchar_t, char_traits<wchar_t>>.....](#)

[12-122 libstdcxx - Class basicifstream<wchar_t, char_traits<wchar_t>> Function Interfaces.....](#)

[12-123 Primary vtable for basicofstream<char, char_traits<char>>.....](#)

[12-124 Secondary vtable for basicofstream<char, char_traits<char>>.....](#)

[12-125 VTT for basicofstream<char, char_traits<char>>.....](#)

[12-126 libstdcxx - Class basicofstream<char, char_traits<char>> Function Interfaces.....](#)

[12-127 Primary vtable for basicofstream<wchar_t, char_traits<wchar_t>>.....](#)

[12-128 Secondary vtable for basic_ofstream<wchar_t, char_traits<wchar_t>>](#).....

[12-129 VTT for basic_ofstream<wchar_t, char_traits<wchar_t>>](#).....

[12-130 libstdcxx - Class basic_ofstream<wchar_t, char_traits<wchar_t>> Function Interfaces](#).....

[12-131 Primary vtable for basic_streambuf<char, char_traits<char>>](#).....

[12-132 typeinfo for basic_streambuf<char, char_traits<char>>](#).....

[12-133 libstdcxx - Class basic_streambuf<char, char_traits<char>> Function Interfaces](#).....

[12-134 Primary vtable for basic_streambuf<wchar_t, char_traits<wchar_t>>](#).....

[12-135 typeinfo for basic_streambuf<wchar_t, char_traits<wchar_t>>](#).....

[12-136 libstdcxx - Class basic_streambuf<wchar_t, char_traits<wchar_t>> Function Interfaces](#).....

[12-137 Primary vtable for basic_filebuf<char, char_traits<char>>](#).....

[12-138 typeinfo for basic_filebuf<char, char_traits<char>>](#).....

[12-139 libstdcxx - Class basic_filebuf<char, char_traits<char>> Function Interfaces](#).....

[12-140 Primary vtable for basic_filebuf<wchar_t, char_traits<wchar_t>>](#).....

[12-141 typeinfo for basic_filebuf<wchar_t, char_traits<wchar_t>>](#).....

[12-142 libstdcxx - Class basic_filebuf<wchar_t, char_traits<wchar_t>> Function Interfaces](#).....

[12-143 typeinfo for ios_base](#).....

[12-144 typeinfo for basic_ios<wchar_t, char_traits<wchar_t>>](#).....

[12-145 typeinfo for ios_base::failure](#).....

[12-146 typeinfo for __timepunct<char>](#).....

[12-147 libstdcxx - Class __timepunct<char> Function Interfaces](#).....

[12-148 typeinfo for __timepunct<wchar_t>](#).....

[12-149 libstdcxx - Class __timepunct<wchar_t> Function Interfaces](#).....

[12-150 typeinfo for messages_base](#).....

[12-151 libstdcxx - Class messages<char> Function Interfaces](#).....

[12-152 libstdcxx - Class messages<wchar_t> Function Interfaces](#).....

[12-153 typeinfo for messages_byname<char>](#).....

[12-154 libstdcxx - Class messages_byname<char> Function Interfaces](#).....

[12-155 typeinfo for messages_byname<wchar_t>](#).....

[12-156 libstdcxx - Class messages_byname<wchar_t> Function Interfaces](#).....

[12-157 typeinfo for numpunct<char>](#).....

[12-158 libstdcxx - Class numpunct<char> Function Interfaces](#).....

[12-159 typeinfo for numpunct<wchar_t>](#).....

[12-160 libstdcxx - Class numpunct<wchar_t> Function Interfaces](#).....

[12-161 typeinfo for numpunct_byname<char>](#).....

[12-162 libstdcxx - Class numpunct_byname<char> Function Interfaces](#).....

[12-163 typeinfo for numpunct_byname<wchar_t>](#).....

[12-164 libstdcxx - Class numpunct_byname<wchar_t> Function Interfaces](#).....

[12-165 typeinfo for codecvt_base](#).....

[12-166 Primary vtable for codecvt<char, char, __mbstate_t>](#).....

[12-167 typeinfo for codecvt<char, char, __mbstate_t>](#).....

[12-168 libstdcxx - Class codecvt<char, char, __mbstate_t> Function Interfaces](#).....

[12-169 Primary vtable for codecvt<wchar_t, char, __mbstate_t>](#).....

[12-170 typeinfo for codecvt<wchar_t, char, __mbstate_t>](#).....

[12-171 libstdcxx - Class codecvt<wchar_t, char, __mbstate_t> Function Interfaces](#).....

[12-172 Primary vtable for codecvt_byname<char, char, __mbstate_t>](#).....

[12-173 typeinfo for codecvt_byname<char, char, __mbstate_t>](#).....

[12-174 libstdcxx - Class codecvt_byname<char, char, __mbstate_t> Function Interfaces](#).....

[12-175 Primary vtable for codecvt_byname<wchar_t, char, __mbstate_t>](#).....

[12-176 typeinfo for codecvt_byname<wchar_t, char, __mbstate_t>](#).....

[12-177 libstdcxx - Class codecvt_byname<wchar_t, char, __mbstate_t> Function Interfaces](#).....

[12-178 typeinfo for collate<char>](#).....

[12-179 libstdcxx - Class collate<char> Function Interfaces](#).....

[12-180 typeinfo for collate<wchar_t>](#).....

[12-181 libstdcxx - Class collate<wchar_t> Function Interfaces](#).....

[12-182 typeinfo for collate byname<char>](#).....

[12-183 libstdcxx - Class collate byname<char> Function Interfaces](#).....

[12-184 typeinfo for collate byname<wchar_t>](#).....

[12-185 libstdcxx - Class collate byname<wchar_t> Function Interfaces](#).....

[12-186 typeinfo for time base](#).....

[12-187 typeinfo for time get byname<char, istreambuf_iterator<char, char_traits<char>>>](#).....

[12-188 libstdcxx - Class time get byname<char, istreambuf_iterator<char, char_traits<char>>> Function Interfaces](#).....

[12-189 typeinfo for time get byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>](#).....

[12-190 libstdcxx - Class time get byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>> Function Interfaces](#).....

[12-191 typeinfo for time put byname<char, ostreambuf_iterator<char, char_traits<char>>>](#).....

[12-192 libstdcxx - Class time put byname<char, ostreambuf_iterator<char, char_traits<char>>> Function Interfaces](#).....

[12-193 typeinfo for time put byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>](#).....

[12-194 libstdcxx - Class time put byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>> Function Interfaces](#).....

[12-195 libstdcxx - Class time get<char, istreambuf_iterator<char, char_traits<char>>> Function Interfaces](#).....

[12-196 libstdcxx - Class time get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>> Function Interfaces](#).....

[12-197 typeinfo for time put<char, ostreambuf_iterator<char, char_traits<char>>>](#).....

[12-198 libstdcxx - Class time put<char, ostreambuf_iterator<char, char_traits<char>>> Function Interfaces](#).....

[12-199 typeinfo for time put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>](#).....

[12-200 libstdcxx - Class time put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>> Function Interfaces](#).....

[12-201 libstdcxx - Class moneypunct<char, false> Function Interfaces](#).....

[12-202 libstdcxx - Class moneypunct<char, true> Function Interfaces](#).....

[12-203 libstdcxx - Class moneypunct<wchar_t, false> Function Interfaces](#).....

[12-204 libstdcxx - Class moneypunct<wchar_t, true> Function Interfaces](#).....

[12-205 typeinfo for moneypunct byname<char, false>](#).....

[12-206 libstdcxx - Class moneypunct byname<char, false> Function Interfaces](#).....

[12-207 typeinfo for moneypunct byname<char, true>](#).....

[12-208 libstdcxx - Class moneypunct byname<char, true> Function Interfaces](#).....

[12-209 typeinfo for moneypunct byname<wchar_t, false>](#).....

[12-210 libstdcxx - Class moneypunct byname<wchar_t, false> Function Interfaces](#).....

[12-211 typeinfo for moneypunct byname<wchar_t, true>](#).....

[12-212 libstdcxx - Class moneypunct byname<wchar_t, true> Function Interfaces](#).....

[12-213 typeinfo for money base](#).....

[12-214 typeinfo for money get<char, istreambuf_iterator<char, char_traits<char>>>](#).....

[12-215 libstdcxx - Class money get<char, istreambuf_iterator<char, char_traits<char>>> Function Interfaces](#).....

[12-216 typeinfo for money get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>](#).....

[12-217 libstdcxx - Class money get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>> Function Interfaces](#).....

[12-218 typeinfo for money put<char, ostreambuf_iterator<char, char_traits<char>>>](#).....

[12-219 libstdcxx - Class money put<char, ostreambuf_iterator<char, char_traits<char>>> Function Interfaces](#).....

[12-220 typeinfo for money put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>](#).....

[12-221 libstdcxx - Class money put<wchar_t, ostreambuf_iterator<wchar_t,](#)

[char_traits<wchar_t>> Function Interfaces](#).....

[12-222 libstdcxx - Class locale Function Interfaces](#).....

[12-223 typeid for locale::facet](#).....

[12-224 typeid for num_get<char, istreambuf_iterator<char, char_traits<char>>>](#).....

[12-225 libstdcxx - Class num_get<char, istreambuf_iterator<char, char_traits<char>>> Function Interfaces](#).....

[12-226 typeid for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>](#).....

[12-227 libstdcxx - Class num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>> Function Interfaces](#).....

[12-228 typeid for num_put<char, ostreambuf_iterator<char, char_traits<char>>>](#).....

[12-229 libstdcxx - Class num_put<char, ostreambuf_iterator<char, char_traits<char>>> Function Interfaces](#).....

[12-230 typeid for num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>](#).....

[12-231 libstdcxx - Class num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>> Function Interfaces](#).....

[12-232 libstdcxx - Class gslice Function Interfaces](#).....

[12-233 libstdcxx - Class basic_file<char> Function Interfaces](#).....

[12-234 libstdcxx - Class valarray<unsigned int> Function Interfaces](#).....

[12-235 libstdcxx - Class gnu_cxx::pool<true> Function Interfaces](#).....

[12-236 libstdcxx - Class gnu_cxx::pool<false> Function Interfaces](#).....

[12-237 libstdcxx - Class gnu_cxx::free_list Function Interfaces](#).....

[12-238 libstdcxx - Class locale::Impl Function Interfaces](#).....

[12-239 libstdcxx - Namespace std Functions Function Interfaces](#).....

[A-1 libc Function Interfaces](#).....

[A-2 libc Data Interfaces](#).....

[A-3 libcrypt Function Interfaces](#).....

[A-4 libdl Function Interfaces](#).....

[A-5 libgcc_s Function Interfaces](#).....

[A-6 libm Function Interfaces](#).....

[A-7 libm Data Interfaces](#).....

[A-8 libpthread Function Interfaces](#).....

[A-9 librt Function Interfaces](#).....

[A-10 libutil Function Interfaces](#).....

Foreword

This is version 5.0 of the Linux Standard Base Core Specification for X86-64. This specification is one of a series of volumes under the collective title *Linux Standard Base*:

- Common
- Core
- Desktop
- Languages
- Imaging

Note that the Core and Desktop volumes consist of a generic volume augmented by an architecture-specific volume.

Status of this Document

This is a released specification, version 5.0. Other documents may supersede or augment this specification.

A list of current released Linux Standard Base (LSB) specifications is available at <http://refspecs.linuxbase.org> (<http://refspecs.linuxbase.org/>).

If you wish to make comments regarding this document in a manner that is tracked by the LSB project, please submit them using our public bug database at <http://bugs.linuxbase.org>. Please enter your feedback, carefully indicating the title of the section for which you are submitting feedback, and the volume and version of the specification where you found the problem, quoting the incorrect text if appropriate. If you are suggesting a new feature, please indicate what the problem you are trying to solve is. That is more important than the solution, in fact.

If you do not have or wish to create a bug database account then you can also e-mail feedback to lsb-discuss@lists.linuxfoundation.org (subscribe (<http://lists.linuxfoundation.org/mailman/listinfo/lsb-discuss>), archives (<http://lists.linuxfoundation.org/pipermail/lsb-discuss/>)), and arrangements will be made to transpose the comments to our public bug database.

Introduction

The LSB defines a binary interface for application programs that are compiled and packaged for LSB-conforming implementations on many different hardware architectures. A binary specification must include information specific to the computer processor architecture for which it is intended. To avoid the complexity of conditional descriptions, the specification has instead been divided into generic parts which are augmented by one of several architecture-specific parts, depending on the target processor architecture; the generic part will indicate when reference must be made to the architecture part, and vice versa.

This document should be used in conjunction with the documents it references. This document enumerates the system components it includes, but descriptions of those components may be included entirely or partly in this document, partly in other documents, or entirely in other reference documents. For example, the section that describes system service routines includes a list of the system routines supported in this interface, formal declarations of the data structures they use that are visible to applications, and a pointer to the underlying referenced specification for information about the syntax and semantics of each call. Only those routines not described in standards referenced by this document, or extensions to those standards, are described in the detail. Information referenced in this way is as much a part of this document as is the information explicitly included here.

The specification carries a version number of either the form $x.y$ or $x.y.z$. This version number carries the following meaning:

1. The first number (x) is the major version number. Versions sharing the same major version number shall be compatible in a backwards direction; that is, a newer version shall be compatible with an older version. Any deletion of a library results in a new major version number. Interfaces marked as deprecated may be removed from the specification at a major version change.
2. The second number (y) is the minor version number. Libraries and individual interfaces may be added, but not removed. Interfaces may be marked as deprecated at a minor version change. Other minor changes may be permitted at the discretion of the LSB workgroup.
3. The third number (z), if present, is the editorial level. Only editorial changes should be included in such versions.

Since this specification is a descriptive Application Binary Interface, and not a source level API specification, it is not possible to make a guarantee of 100% backward compatibility between major releases. However, it is the intent that those parts of the binary interface that are visible in the source level API will remain backward compatible from version to version, except where a feature marked as "Deprecated" in one release may be removed from a future release. Implementors are strongly encouraged to make use of symbol versioning to permit simultaneous support of applications conforming to different releases of this specification.

LSB is a trademark of the Linux Foundation. Developers of applications or implementations interested in using the trademark should see the Linux Foundation Certification Policy for details.

I Introductory Elements

1 Scope

1.1 General

The Linux Standard Base (LSB) defines a system interface for compiled applications and a minimal environment for support of installation scripts. Its purpose is to enable a uniform industry standard environment for high-volume applications conforming to the LSB.

These specifications are composed of two basic parts: a common part describing those parts of the interface that remain constant across all implementations of the LSB, and an architecture-specific part describing the parts of the interface that vary by processor architecture. Together, the common part and the relevant architecture-specific part for a single hardware architecture provide a complete interface specification for compiled application programs on systems that share a common hardware architecture.

The LSB contains both a set of Application Program Interfaces (APIs) and Application Binary Interfaces (ABIs). APIs may appear in the source code of portable applications, while the compiled binary of that application may use the larger set of ABIs. A conforming implementation provides all of the ABIs listed here. The compilation system may replace (e.g. by macro definition) certain APIs with calls to one or more of the underlying binary interfaces, and may insert calls to binary interfaces as needed.

The LSB is primarily a binary interface definition. Not all of the source level APIs available to applications may be contained in this specification.

1.2 Module Specific Scope

This is the X86-64 architecture specific part of the Core module of the Linux Standard Base (LSB). This part supplements the common part of the LSB Core module with those interfaces that differ between architectures.

This part should be used in conjunction with LSB Core - Generic, the common part. Whenever a section of the common part is supplemented by architecture-specific information, the common part includes a reference to the architecture-specific part. This part may also contain additional information that is not referenced in the common part.

Interfaces described in this part of the LSB Core Specification are mandatory except where explicitly listed otherwise. Interfaces described in the LSB Core module are supplemented by other LSB modules. All other modules depend on the presence of LSB Core.

2 References

2.1 Normative References

The following specifications are incorporated by reference into this specification. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced specification (including any amendments) applies.

Note: Where copies of a referenced specification are available on the World Wide Web, a Uniform Resource Locator (URL) is given, for informative purposes only. Such URL might at any given time resolve to a more recent copy of the specification, or be out of date (not resolve). Reference copies of specifications at the revision level indicated may be found at the Linux Foundation's Reference Specifications (<http://refspecs.linuxbase.org>) site.

Table 2-1 Normative References

Name	Title	URL
LSB Core - Generic	Linux Standard Base - Core Specification - Generic	http://www.linuxbase.org/spec/
AMD64 Architecture Programmer's Manual, Volume 1	AMD64 Architecture Programmer's Manual, Volume 1: Application Programming 24592 3.08	http://www.amd.com/us-en/Processors/DevelopWithAMD/
AMD64 Architecture Programmer's Manual, Volume 2	AMD64 Architecture Programmer's Manual, Volume 2: System Programming 24593 3.08	http://www.amd.com/us-en/Processors/DevelopWithAMD/
AMD64 Architecture Programmer's Manual, Volume 3	AMD64 Architecture Programmer's Manual, Volume 3: General Purpose and System Instructions 24594 3.03	http://www.amd.com/us-en/Processors/DevelopWithAMD/
AMD64 Architecture Programmer's Manual, Volume 4	AMD64 Architecture Programmer's Manual, Volume 4: 128-bit Media Instructions 26568 3.04	http://www.amd.com/us-en/Processors/DevelopWithAMD/
AMD64 Architecture Programmer's Manual, Volume 5	AMD64 Architecture Programmer's Manual, Volume 5: 64-bit Media and x87 Floating-Point Instructions 26569 3.03	http://www.amd.com/us-en/Processors/DevelopWithAMD/
Filesystem Hierarchy Standard	Filesystem Hierarchy Standard (FHS) 3.0	http://refspecs.linuxbase.org/fhs
ISO C (1999)	ISO/IEC 9899:1999 - Programming Languages -- C	
ISO/IEC 14882: 2003 C++ Language	ISO/IEC 14882: 2003 Programming languages --C++	
Itanium™ C++ ABI	Itanium™ C++ ABI (Revision 1.86)	http://refspecs.linuxfoundation.org/cxxabi-1.86.html
Large File Support	Large File Support	http://www.UNIX-systems.org/version2/wha

		tsnew/lfs20mar.html
Libncursesw API	Libncursesw API	http://invisible-island.net/ncurses/man/ncurses.3x.html
Libncursesw Placeholder	Libncursesw Specification Placeholder	http://refspecs.linux-foundation.org/libncursesw/libncurses.html
POSIX 1003.1-2001 (ISO/IEC 9945-2003)	ISO/IEC 9945-1:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 1: Base Definitions ISO/IEC 9945-2:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 2: System Interfaces ISO/IEC 9945-3:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 3: Shell and Utilities ISO/IEC 9945-4:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 4: Rationale Including Technical Cor. 1: 2004	http://www.unix.org/version3/
POSIX 1003.1-2008 (ISO/IEC 9945-2009)	Portable Operating System Interface (POSIX®) 2008 Edition / The Open Group Technical Standard Base Specifications, Issue 7	http://www.unix.org/version4/
SUSv2	CAE Specification, January 1997, System Interfaces and Headers (XSH), Issue 5 (ISBN: 1-85912-181-0, C606)	http://www.opengroup.org/publications/catalog/un.htm
SVID Issue 3	American Telephone and Telegraph Company, System V Interface Definition, Issue 3; Morristown, NJ, UNIX Press, 1989. (ISBN 0201566524)	
SVID Issue 4	System V Interface Definition, Fourth Edition	http://refspecs.linuxfoundation.org/svid4/
System V ABI	System V Application Binary Interface, Edition 4.1	http://www.sco.com/developers/devspecs/gabi41.pdf
System V ABI Update	System V Application	http://www.sco.com/develop

	Binary Interface - DRAFT - 17 December 2003	opers/gabi/2003-12-17/contents.html
System V Application Binary Interface AMD64 Architecture Processor Supplement	System V Application Binary Interface AMD64 Architecture Processor Supplement, Draft Version 0.95	http://refspecs.linux-foundation.org/elf/x86_64-abi-0.95.pdf
X/Open Curses, Issue 7	X/Open Curses, Issue 7 (ISBN: 1-931624-83-6, The Open Group, November 2009)	https://www2.opengroup.org/ogsys/catalog/C094

2.2 Informative References/Bibliography

The documents listed below provide essential background information to implementors of this specification. These references are included for information only, and do not represent normative parts of this specification.

Table 2-2 Other References

Name	Title	URL
DWARF Debugging Information Format, Version 4	DWARF Debugging Information Format, Version 4 (June 10, 2010)	http://www.dwarfstd.org/doc/DWARF4.pdf
IEC 60559/IEEE 754 Floating Point	IEC 60559:1989 Binary floating-point arithmetic for microprocessor systems	http://www.ieee.org/
ISO/IEC TR14652	ISO/IEC Technical Report 14652:2002 Specification method for cultural conventions	
ITU-T V.42	International Telecommunication Union Recommendation V.42 (2002): Error-correcting procedures for DCEs using asynchronous-to-synchronous conversion ITUV	http://www.itu.int/rec/rec-ommendation.asp?type=folders&lang=e&parent=T-REC-V.42
Linux Globalization Specification	LI18NUNIX 2000 Globalization Specification, Version 1.0 with Amendment 4	http://www.openl18n.org/docs/html/LI18NUNIX-2000-amd4.htm
Linux Allocated Device Registry	LINUX ALLOCATED DEVICES	http://www.lanana.org/docs/device-list/devices-2.6+.txt
Linux Assigned Names And Numbers Authority	Linux Assigned Names And Numbers Authority	http://www.lanana.org/
Mozilla's NSS SSL Reference	Mozilla's NSS SSL Reference	http://www.mozilla.org/projects/security/pki/nss/ref/ssl/
NSPR Reference	Mozilla's NSPR	http://refspecs.linuxfound

	Reference	ation.org/NSPR_API_Reference/NSPR_API.html
PAM	Open Software Foundation, Request For Comments: 86.0 , October 1995, V. Samar & R.Schemers (SunSoft)	http://www.opengroup.org/tech/rfc/mirror-rfc/rfc86.0.txt
RFC 1321: The MD5 Message-Digest Algorithm	IETF RFC 1321: The MD5 Message-Digest Algorithm	http://www.ietf.org/rfc/rfc1321.txt
RFC 1833: Binding Protocols for ONC RPC Version 2	IETF RFC 1833: Binding Protocols for ONC RPC Version 2	http://www.ietf.org/rfc/rfc1833.txt
RFC 1950: ZLIB Compressed Data Format Specification	IETF RFC 1950: ZLIB Compressed Data Format Specification	http://www.ietf.org/rfc/rfc1950.txt
RFC 1951: DEFLATE Compressed Data Format Specification	IETF RFC 1951: DEFLATE Compressed Data Format Specification version 1.3	http://www.ietf.org/rfc/rfc1951.txt
RFC 1952: GZIP File Format Specification	IETF RFC 1952: GZIP file format specification version 4.3	http://www.ietf.org/rfc/rfc1952.txt
RFC 2440: OpenPGP Message Format	IETF RFC 2440: OpenPGP Message Format	http://www.ietf.org/rfc/rfc2440.txt
RFC 2821: Simple Mail Transfer Protocol	IETF RFC 2821: Simple Mail Transfer Protocol	http://www.ietf.org/rfc/rfc2821.txt
RFC 2822: Internet Message Format	IETF RFC 2822: Internet Message Format	http://www.ietf.org/rfc/rfc2822.txt
RFC 5531/4506 RPC & XDR	IETF RFC 5531 & 4506	http://www.ietf.org/
RFC 791: Internet Protocol	IETF RFC 791: Internet Protocol Specification	http://www.ietf.org/rfc/rfc791.txt
RPM Package Format	RPM Package Format V3.0	http://www.rpm.org/max-rpm/s1-rpm-file-format-rpm-file-format.html
zlib Manual	zlib 1.2 Manual	http://www.gzip.org/zlib/

3 Requirements

3.1 Relevant Libraries

The libraries listed in [Table 3-1](#) shall be available on x86-64 Linux Standard Base systems, with the specified runtime names. These names override or supplement the names specified in the generic LSB (LSB Core - Generic) specification. The specified program interpreter, referred to as `proginterp` in this table, shall be used to load the shared libraries specified by `DT_NEEDED` entries at run time.

Table 3-1 Standard Library Names

Library	Runtime Name
libc	libc.so.6
libcrypt	libcrypt.so.1
libdl	libdl.so.2
libgcc_s	libgcc_s.so.1
libm	libm.so.6
libncurses	libncurses.so.5
libncursesw	libncursesw.so.5
libpthread	libpthread.so.0
libstdcxx	libstdc++.so.6
libutil	libutil.so.1
libz	libz.so.1
proginterp	/lib64/ld-lsb-x86-64.so.3

These libraries will be in an implementation-defined directory which the dynamic linker shall search by default.

3.2 LSB Implementation Conformance

A conforming implementation is necessarily architecture specific, and must provide the interfaces specified by both the generic LSB Core specification (LSB Core - Generic) and the relevant architecture specific part of the LSB Core Specification.

Rationale: An implementation must provide *at least* the interfaces specified in these specifications. It may also provide additional interfaces.

A conforming implementation shall satisfy the following requirements:

- A processor architecture represents a family of related processors which may not have identical feature sets. The architecture specific parts of the LSB Core Specification that supplement this specification for a given target processor architecture describe a minimum acceptable processor. The implementation shall provide all features of this processor, whether in hardware or through emulation transparent to the application.
- The implementation shall be capable of executing compiled applications having the format and using the system interfaces described in this specification.
- The implementation shall provide libraries containing the interfaces specified by this specification, and shall provide a dynamic linking mechanism that allows these interfaces to be attached to applications at runtime. All the interfaces shall behave as specified in this specification.
- The map of virtual memory provided by the implementation shall conform to the requirements of this specification.

- The implementation's low-level behavior with respect to function call linkage, system traps, signals, and other such activities shall conform to the formats described in this specification.
- The implementation shall provide all of the mandatory interfaces in their entirety.
- The implementation may provide one or more of the optional interfaces. Each optional interface that is provided shall be provided in its entirety. The product documentation shall state which optional interfaces are provided.
- The implementation shall provide all files and utilities specified as part of this specification in the format defined here and in other documents normatively included by reference. All commands and utilities shall behave as required by this specification. The implementation shall also provide all mandatory components of an application's runtime environment that are included or referenced in this specification.
- The implementation, when provided with standard data formats and values at a named interface, shall provide the behavior defined for those values and data formats at that interface. However, a conforming implementation may consist of components which are separately packaged and/or sold. For example, a vendor of a conforming implementation might sell the hardware, operating system, and windowing system as separately packaged items.
- The implementation may provide additional interfaces with different names. It may also provide additional behavior corresponding to data values outside the standard ranges, for standard named interfaces.

3.3 LSB Application Conformance

A conforming application containing object files is necessarily architecture specific, and must conform to both the generic LSB Core specification (LSB Core - Generic) and the relevant architecture specific part of the LSB Core Specification. A conforming application which contains no object files may be architecture neutral. Architecture neutral applications shall conform only to the requirements of the generic LSB Core specification (LSB Core - Generic).

A conforming application shall satisfy the following requirements:

- Executable files shall be either object files in the format defined in the Object Format section of this specification, or script files in a scripting language where the interpreter is required by this specification.
- Object files shall participate in dynamic linking as defined in the Program Loading and Linking section of this specification.
- Object files shall employ only the instructions, traps, and other low-level facilities defined as being for use by applications in the Low-Level System Information section of this specification
- If the application requires any optional interface defined in this specification in order to be installed or to execute successfully, the requirement for that optional interface shall be stated in the application's documentation.
- The application shall not use any interface or data format that is not required to be provided by a conforming implementation, unless such an interface or data format is supplied by another application through direct invocation of that application during execution. The other application must also be a conforming application, and the use of such interface or data format, as well as its source (in other words, the other conforming application), shall be identified in the documentation of the application.
- The application shall not use any values for a named interface that are reserved for vendor extensions.

A strictly conforming application shall not require or use any interface, facility, or implementation-defined extension not defined in this specification in order to be installed or to execute successfully.

4 Terms and Definitions

For the purposes of this document, the terms given in *ISO/IEC Directives, Part 2, Annex H* and the following apply.

archLSB

Some LSB specification documents have both a generic, architecture-neutral part and an architecture-specific part. The latter describes elements whose definitions may be unique to a particular processor architecture. The term archLSB may be used in the generic part to refer to the corresponding section of the architecture-specific part.

Binary Standard, ABI

The total set of interfaces that are available to be used in the compiled binary code of a conforming application, including the run-time details such as calling conventions, binary format, C++ name mangling, etc.

Implementation-defined

Describes a value or behavior that is not defined by this document but is selected by an implementor. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence of the value or behavior. An application that relies on such a value or behavior cannot be assured to be portable across conforming implementations. The implementor shall document such a value or behavior so that it can be used correctly by an application.

Shell Script

A file that is read by an interpreter (e.g., awk). The first line of the shell script includes a reference to its interpreter binary.

Source Standard, API

The total set of interfaces that are available to be used in the source code of a conforming application. Due to translations, the Binary Standard and the Source Standard may contain some different interfaces.

Undefined

Describes the nature of a value or behavior not defined by this document which results from use of an invalid program construct or invalid data input. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

Unspecified

Describes the nature of a value or behavior not specified by this document which results from use of a valid program construct or valid data input. The value or behavior may vary among implementations that conform to this document. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

In addition, for the portions of this specification which build on IEEE Std 1003.1-2001, the definitions given in *IEEE Std 1003.1-2001, Base Definitions, Chapter 3* apply.

5 Documentation Conventions

Throughout this document, the following typographic conventions are used:

`function()`

the name of a function

command

the name of a command or utility

CONSTANT

a constant value

parameter

a parameter

variable

a variable

Throughout this specification, several tables of interfaces are presented. Each entry in these tables has the following format:

name

the name of the interface

(symver)

An optional symbol version identifier, if required.

[*refno*]

A reference number indexing the table of referenced specifications that follows this table.

For example,

<code>forkpty(GLIBC_2.0) [SUSv4]</code>

refers to the interface named `forkpty()` with symbol version `GLIBC_2.0` that is defined in the reference indicated by the tag `SUSv4`.

Note: For symbols with versions which differ between architectures, the symbol versions are defined in the architecture specific parts of of this module specification only. In the generic part, they will appear without symbol versions.

II Executable and Linking Format (ELF)

6 Introduction

Executable and Linking Format (ELF) defines the object format for compiled applications. This specification supplements the information found in [System V ABI Update](#) and [System V Application Binary Interface AMD64 Architecture Processor Supplement](#), and is intended to document additions made since the publication of that document.

7 Low Level System Information

7.1 Machine Interface

7.1.1 Processor Architecture

The AMD64 Architecture is specified by the following documents

- [AMD64 Architecture Programmer's Manual, Volume 1](#)
- [AMD64 Architecture Programmer's Manual, Volume 2](#)
- [AMD64 Architecture Programmer's Manual, Volume 3](#)
- [AMD64 Architecture Programmer's Manual, Volume 4](#)
- [AMD64 Architecture Programmer's Manual, Volume 5](#)
- [System V Application Binary Interface AMD64 Architecture Processor Supplement](#)

Applications conforming to this specification must provide feedback to the user if a feature that is required for correct execution of the application is not present. Applications conforming to this specification should attempt to execute in a diminished capacity if a required instruction set feature is not present. In particular, applications should not rely on the availability of the 3DNow!™ technology. In addition, a conforming application shall not use any instruction from [Table 7-1](#).

Note: While this specification carries the attribution "AMD64", it is intended to apply to the entire x86_64 set of processors, including those based on Intel® 64 Architecture. However, this specification defers to the AMD64 architecture specifications listed above.

Table 7-1 Non Conforming Instructions

LAHF	SAHF
SYSCALL	SYSRET
SYSENTER	SYSEXIT
CMPXCHG16B	FFXSR

Conforming applications may use only instructions which do not require elevated privileges.

Conforming applications shall not invoke the implementations underlying system call interface directly. The interfaces in the implementation base libraries shall be used instead.

Rationale: Implementation-supplied base libraries may use the system call interface but applications must not assume any particular operating system or kernel version is present.

This specification does not provide any performance guarantees of a conforming system. A system conforming to this specification may be implemented in either hardware or software.

7.1.2 Data Representation

7.1.2.1 Introduction

LSB-conforming applications shall use the data representation as defined in Section 3.1.2 of [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

Note: The [System V Application Binary Interface AMD64 Architecture Processor Supplement](#) specification is itself layered on top of the System V Application Binary Interface - Intel386™ Architecture Processor Supplement.

7.1.2.2 Byte Ordering

LSB-conforming applications shall use the byte ordering defined in Section 3.1.2 of [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.1.2.3 Fundamental Types

LSB-conforming applications shall use only the fundamental types described in Section 3.1.2 of [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.1.2.4 Aggregates and Unions

LSB-conforming applications shall use alignment for aggregates and unions as described in Section 3.1.2 of [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.1.2.5 Bit Fields

LSB-conforming applications utilizing bit-fields shall follow the requirements of Section 3.1.2 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.2 Function Calling Sequence

7.2.1 Introduction

LSB-conforming applications shall use only the following features of the function calling sequence as defined in Section 3.2 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.2.2 Registers

LSB-conforming applications shall use only the registers described in Section 3.2.1 (Registers and the Stack Frame) of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.2.3 Floating Point Registers

LSB-conforming applications shall use only the floating point registers described in Section 3.2.1 (Registers and the Stack Frame) of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.2.4 Stack Frame

LSB-conforming applications shall use stack frames as described in Section 3.2.2 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.2.5 Arguments

LSB-conforming applications shall pass parameters to functions as described in Section 3.2.3 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.2.6 Return Values

Values are returned from functions as described in Section 3.3.2 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.3 Operating System Interface

LSB-conforming applications shall use only the following features of the Operating Sys-

tem Interfaces as defined in Section 3.3 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.3.1 Exception Interface

Synchronous and floating point or coprocessor exceptions shall behave as described in Section 3.3.1 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.3.2 Virtual Address Space

LSB-Conforming applications shall use only the virtual address space described in Section 3.3.2 and 3.3.4 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#). Virtual memory page sizes shall be subject to the limitations described in Section 3.3.3 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.4 Process Initialization

LSB-conforming applications shall use only the following features of the Process Initialization as defined in Section 3.4 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.4.1 Special Registers

During process initialization, the special registers shall be initialized as described in Section 3.4.1 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.4.2 Process Stack (on entry)

The process stack shall be initialized as described in Section 3.4.1 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.4.3 Auxiliary Vector

The auxiliary vector shall be initialized as described in Section 3.4.2 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.5 Coding Examples

LSB-conforming applications may use the coding examples given in Section 3.5 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#) to guide implementation of fundamental operations in the following areas.

7.5.1 Code Model Overview/Architecture Constraints

Section 3.5.1 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#) describes a number of code models. LSB-Conforming applications may use any of these models except the Kernel and Large code models.

7.5.2 Position-Independent Function Prologue

LSB-conforming applications may follow the position-independent function prologue example in Section 3.5.3 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.5.3 Data Objects

LSB-conforming applications may follow the data objects examples in Section 3.5.4 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.5.4 Function Calls

LSB-conforming applications may follow the function call examples in Section 3.5.5 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#). See Chapter 3 of [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.5.5 Branching

LSB-conforming applications may follow the branching examples in Section 3.5.6 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.6 C Stack Frame

7.6.1 Variable Argument List

LSB-Conforming applications shall only use variable arguments to functions in the manner described in Section 3.5.7 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

7.7 Debug Information

LSB-Conforming applications may include DWARF debugging information. The DWARF Release Number and Register Number Mapping shall be as described in Section 3.6 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

8 Object Format

8.1 Introduction

LSB-conforming implementations shall support the Executable and Linking Format (ELF) object file format, as defined by the following documents:

- [System V ABI](#)
- [System V ABI Update](#)
- [System V Application Binary Interface AMD64 Architecture Processor Supplement](#)
- [LSB Core - Generic](#)
- this document

8.2 ELF Header

8.2.1 Machine Information

LSB-conforming applications shall identify the Machine Information as defined in Section 4.1.1 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

8.3 Sections

8.3.1 Introduction

In addition to the requirements for ELF sections described in the generic LSB Core specification, conforming implementations shall support architecture specific sections as described below.

Note: The [System V Application Binary Interface AMD64 Architecture Processor Supplement](#) specifies some architecture specific section flags and section types that are not required by LSB-conforming systems.

8.3.2 Special Sections

The following architecture-specific sections are defined in the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

Table 8-1 ELF Special Sections

Name	Type	Attributes
.got	SHT_PROGBITS	SHF_ALLOC+SHF_WRITE
.plt	SHT_PROGBITS	SHF_ALLOC+SHF_EXECINSTR

.got

This section holds the global offset table.

.plt

This section holds the procedure linkage table.

Note: Since LSB-conforming implementations are not required to support the large code model, it is not necessary for them to provide support for the additional special sections for the large code model described in the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

Also, the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#) specifies a section `.eh_frame`, with a type of `SHT_AMD64_UNWIND`. This section is described in the generic LSB-Core specification, but with type `SHT_PROGBITS`. This specification does not require support for the `SHT_AMD64_UNWIND` section type.

8.3.3 Additional Special Sections

The following additional sections are defined here.

Table 8-2 Additional Special Sections

Name	Type	Attributes
<code>.rela.dyn</code>	<code>SHT_RELA</code>	<code>SHF_ALLOC</code>
<code>.rela.plt</code>	<code>SHT_RELA</code>	<code>SHF_ALLOC</code>

`.rela.dyn`

This section holds RELA type relocation information for all sections of a shared library except the PLT.

`.rela.plt`

This section holds RELA type relocation information for the PLT section of a shared library or dynamically linked application.

8.4 Symbol Table

LSB-conforming applications shall use Symbol Tables as defined in Section 4.3 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

8.5 Relocation

LSB-conforming implementation shall support the required relocation types defined in Section 4.4.1 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

Note: Since LSB-conforming implementations are not required to support the large code model, it is not necessary for them to provide support for the additional relocation types for the large code model described in the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

9 Program Loading and Dynamic Linking

9.1 Introduction

LSB-conforming implementations shall support the object file information and system actions that create running programs as specified in the [System V ABI](#) , [System V ABI Update](#) , [System V Application Binary Interface AMD64 Architecture Processor Supplement](#) and as supplemented by the generic LSB specification and [LSB Core - Generic](#).

9.2 Program Header

LSB-conforming implementations are not required to support the additional types and flags for this architecture as defined in Section 5.1 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

Note: The [System V Application Binary Interface AMD64 Architecture Processor Supplement](#) specification is itself layered on top of the System V Application Binary Interface - Intel386™ Architecture Processor Supplement. As such, the requirements of that specification are still requirements of this specification.

9.3 Program Loading

LSB-conforming implementations shall map file pages to virtual memory pages as described in Section 5.1 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

9.4 Dynamic Linking

9.4.1 Introduction

LSB-conforming implementations shall provide dynamic linking as specified in Section 5.2 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#), except as described in the following sections.

Note: Since LSB-conforming implementations are not required to support the large model, support for dynamic linking of large model code is not required.

9.4.2 Dynamic Section

Dynamic section entries give information to the dynamic linker. The following dynamic entry types shall be supported:

DT_JMPREL

This entry is associated with a table of relocation entries for the procedure linkage table. This entry is mandatory both for executable and shared object files

DT_PLTGOT

This entry's `d_ptr` member gives the address of the first byte in the procedure linkage table

DT_RELACOUNT

The number of relative relocations in `.rela.dyn`

9.4.3 Global Offset Table

LSB-conforming implementations shall support a Global Offset Table as described in Section 5.2 of the [System V Application Binary Interface AMD64 Architecture](#)

[Processor Supplement](#).

9.4.4 Function Addresses

Function addresses shall behave as described in Section 5.2 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

9.4.5 Procedure Linkage Table

LSB-conforming implementations shall support a Procedure Linkage Table as described in Section 5.2 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

9.4.6 Initialization and Termination Functions

LSB-conforming implementations shall support initialization and termination functions as specified in Section 5.2.2 of the [System V Application Binary Interface AMD64 Architecture Processor Supplement](#).

III Base Libraries

10 Libraries

An LSB-conforming implementation shall support base libraries which provide interfaces for accessing the operating system, processor and other hardware in the system.

Only interfaces and interface details which are unique to the X86-64 platform are defined here. This section should be used in conjunction with the corresponding section of LSB Core - Generic.

10.1 Program Interpreter/Dynamic Linker

The Program Interpreter shall be [/lib64/ld-1sb-x86-64.so.3](#).

10.2 Interfaces for libc

[Table 10-1](#) defines the library name and shared object name for the libc library

Table 10-1 libc Definition

Library:	libc
SONAME:	libc.so.6

The behavior of the interfaces in this library is specified by the following specifications:

- [LFS] [Large File Support](#)
- [LSB] [LSB Core - Generic](#)
- [RPC + XDR] [RFC 5531/4506 RPC & XDR](#)
- [SUSv2] [SUSv2](#)
- [SUSv3] [POSIX 1003.1-2001 \(ISO/IEC 9945-2003\)](#)
- [SUSv4] [POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#)
- [SVID.4] [SVID Issue 4](#)

10.2.1 RPC

10.2.1.1 Interfaces for RPC

An LSB conforming implementation shall provide the architecture specific functions for RPC specified in [Table 10-2](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-2 libc - RPC Function Interfaces

authnone_create(GLIBC_2.2.5) [SVID.4]	callrpc(GLIBC_2.2.5) [RPC + XDR]	clnt_create(GLIBC_2.2.5) [SVID.4]	clnt_pcreateerror(GLIBC_2.2.5) [SVID.4]
clnt_permpno(GLIBC_2.2.5) [SVID.4]	clnt_perror(GLIBC_2.2.5) [SVID.4]	clnt_spcreateerror(GLIBC_2.2.5) [SVID.4]	clnt_spermpno(GLIBC_2.2.5) [SVID.4]
clnt_sperror(GLIBC_2.2.5) [SVID.4]	clntraw_create(GLIBC_2.2.5) [RPC + XDR]	clnttcp_create(GLIBC_2.2.5) [RPC + XDR]	clntudp_bufcreate(GLIBC_2.2.5) [RPC + XDR]
clntudp_create(GLIBC_2.2.5) [RPC + XDR]	key_decryptsession(GLIBC_2.2.5) [SVID.4]	pmap_getport(GLIBC_2.2.5) [LSB]	pmap_set(GLIBC_2.2.5) [LSB]
pmap_unset(GLIBC_2.2.5) [LSB]	svc_getreqset(GLIBC_2.2.5) [SVID.4]	svc_register(GLIBC_2.2.5) [LSB]	svc_run(GLIBC_2.2.5) [LSB]
svc_sendreply(GLIBC_2.2.5) [LSB]	svcerr_auth(GLIBC_2.2.5) [SVID.4]	svcerr_decode(GLIBC_2.2.5)	svcerr_noproc(GLIBC_2.2.5)

		[SVID.4]	[SVID.4]
svcerr_noprogram(GLIBC_2.2.5) [SVID.4]	svcerr_progvers(GLIBC_2.2.5) [SVID.4]	svcerr_systemerr(GLIBC_2.2.5) [SVID.4]	svcerr_weakauth(GLIBC_2.2.5) [SVID.4]
svcf_create(GLIBC_2.2.5) [RPC + XDR]	svcrw_create(GLIBC_2.2.5) [RPC + XDR]	svctcp_create(GLIBC_2.2.5) [LSB]	svcudp_create(GLIBC_2.2.5) [LSB]
xdr_accepted_reply(GLIBC_2.2.5) [SVID.4]	xdr_array(GLIBC_2.2.5) [SVID.4]	xdr_bool(GLIBC_2.2.5) [SVID.4]	xdr_bytes(GLIBC_2.2.5) [SVID.4]
xdr_callhdr(GLIBC_2.2.5) [SVID.4]	xdr_callmsg(GLIBC_2.2.5) [SVID.4]	xdr_char(GLIBC_2.2.5) [SVID.4]	xdr_double(GLIBC_2.2.5) [SVID.4]
xdr_enum(GLIBC_2.2.5) [SVID.4]	xdr_float(GLIBC_2.2.5) [SVID.4]	xdr_free(GLIBC_2.2.5) [SVID.4]	xdr_int(GLIBC_2.2.5) [SVID.4]
xdr_long(GLIBC_2.2.5) [SVID.4]	xdr_opaque(GLIBC_2.2.5) [SVID.4]	xdr_opaque_auth(GLIBC_2.2.5) [SVID.4]	xdr_pointer(GLIBC_2.2.5) [SVID.4]
xdr_reference(GLIBC_2.2.5) [SVID.4]	xdr_rejected_reply(GLIBC_2.2.5) [SVID.4]	xdr_replymsg(GLIBC_2.2.5) [SVID.4]	xdr_short(GLIBC_2.2.5) [SVID.4]
xdr_string(GLIBC_2.2.5) [SVID.4]	xdr_u_char(GLIBC_2.2.5) [SVID.4]	xdr_u_int(GLIBC_2.2.5) [LSB]	xdr_u_long(GLIBC_2.2.5) [SVID.4]
xdr_u_short(GLIBC_2.2.5) [SVID.4]	xdr_union(GLIBC_2.2.5) [SVID.4]	xdr_vector(GLIBC_2.2.5) [SVID.4]	xdr_void(GLIBC_2.2.5) [SVID.4]
xdr_wrapstring(GLIBC_2.2.5) [SVID.4]	xdrmem_create(GLIBC_2.2.5) [SVID.4]	xdrrec_create(GLIBC_2.2.5) [SVID.4]	xdrrec_endofrecord(GLIBC_2.2.5) [RPC + XDR]
xdrrec_eof(GLIBC_2.2.5) [SVID.4]	xdrrec_skiprecord(GLIBC_2.2.5) [RPC + XDR]	xdrstdio_create(GLIBC_2.2.5) [LSB]	

An LSB conforming implementation shall provide the architecture specific deprecated functions for RPC specified in Table 10-3, with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 10-3 libc - RPC Deprecated Function Interfaces

key_decryptsession(GLIBC_2.2.5) [SVID.4]			
--	--	--	--

10.2.2 Epoll

10.2.2.1 Interfaces for Epoll

No external functions are defined for libc - Epoll in this part of the specification. See also the generic specification.

10.2.3 System Calls

10.2.3.1 Interfaces for System Calls

An LSB conforming implementation shall provide the architecture specific functions for System Calls specified in [Table 10-4](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-4 libc - System Calls Function Interfaces

__fxstat(GLIBC_2.2.5) [LSB]	__getpgid(GLIBC_2.2.5) [LSB]	__lxstat(GLIBC_2.2.5) [LSB]	__xmknod(GLIBC_2.2.5) [LSB]
__xstat(GLIBC_2.2.5) [LSB]	access(GLIBC_2.2.5) [SUSv4]	acct(GLIBC_2.2.5) [LSB]	alarm(GLIBC_2.2.5) [SUSv4]
backtrace(GLIBC_2.2.5) [LSB]	backtrace_symbols(GLIBC_2.2.5) [LSB]	backtrace_symbols_fd(GLIBC_2.2.5) [LSB]	brk(GLIBC_2.2.5) [SUSv2]
chdir(GLIBC_2.2.5) [SUSv4]	chmod(GLIBC_2.2.5) [SUSv4]	chown(GLIBC_2.2.5) [SUSv4]	chroot(GLIBC_2.2.5) [SUSv2]
clock(GLIBC_2.2.5) [SUSv4]	close(GLIBC_2.2.5) [SUSv4]	closedir(GLIBC_2.2.5) [SUSv4]	creat(GLIBC_2.2.5) [SUSv4]
dup(GLIBC_2.2.5) [SUSv4]	dup2(GLIBC_2.2.5) [SUSv4]	execl(GLIBC_2.2.5) [SUSv4]	execle(GLIBC_2.2.5) [SUSv4]
execlp(GLIBC_2.2.5) [SUSv4]	execv(GLIBC_2.2.5) [SUSv4]	execve(GLIBC_2.2.5) [SUSv4]	execvp(GLIBC_2.2.5) [SUSv4]
exit(GLIBC_2.2.5) [SUSv4]	fchdir(GLIBC_2.2.5) [SUSv4]	fchmod(GLIBC_2.2.5) [SUSv4]	fchown(GLIBC_2.2.5) [SUSv4]
fcntl(GLIBC_2.2.5) [LSB]	fdatasync(GLIBC_2.2.5) [SUSv4]	fexecve(GLIBC_2.2.5) [SUSv4]	flock(GLIBC_2.2.5) [LSB]
fork(GLIBC_2.2.5) [SUSv4]	fstatfs(GLIBC_2.2.5) [LSB]	fstatvfs(GLIBC_2.2.5) [SUSv4]	fsync(GLIBC_2.2.5) [SUSv4]
ftime(GLIBC_2.2.5) [SUSv3]	ftruncate(GLIBC_2.2.5) [SUSv4]	getcontext(GLIBC_2.2.5) [SUSv3]	getdtablesize(GLIBC_2.2.5) [LSB]
getegid(GLIBC_2.2.5) [SUSv4]	geteuid(GLIBC_2.2.5) [SUSv4]	getgid(GLIBC_2.2.5) [SUSv4]	getgroups(GLIBC_2.2.5) [SUSv4]
getitimer(GLIBC_2.2.5) [SUSv4]	getloadavg(GLIBC_2.2.5) [LSB]	getpagesize(GLIBC_2.2.5) [LSB]	getpgid(GLIBC_2.2.5) [SUSv4]
getpgrp(GLIBC_2.2.5) [SUSv4]	getpid(GLIBC_2.2.5) [SUSv4]	getppid(GLIBC_2.2.5) [SUSv4]	getpriority(GLIBC_2.2.5) [SUSv4]
getrlimit(GLIBC_2.2.5) [LSB]	getrusage(GLIBC_2.2.5) [SUSv4]	getsid(GLIBC_2.2.5) [SUSv4]	getuid(GLIBC_2.2.5) [SUSv4]
getwd(GLIBC_2.2.5) [SUSv3]	initgroups(GLIBC_2.2.5) [LSB]	ioctl(GLIBC_2.2.5) [LSB]	ioperm(GLIBC_2.2.5) [LSB]
iopl(GLIBC_2.2.5) [LSB]	kill(GLIBC_2.2.5) [LSB]	killpg(GLIBC_2.2.5) [SUSv4]	lchown(GLIBC_2.2.5) [SUSv4]
link(GLIBC_2.2.5) [LSB]	lockf(GLIBC_2.2.5) [SUSv4]	lseek(GLIBC_2.2.5) [SUSv4]	mkdir(GLIBC_2.2.5) [SUSv4]
mkfifo(GLIBC_2.2.5) [SUSv4]	mlock(GLIBC_2.2.5) [SUSv4]	mlockall(GLIBC_2.2.5) [SUSv4]	mmap(GLIBC_2.2.5) [SUSv4]
mprotect(GLIBC_2.2.5) [SUSv4]	mremap(GLIBC_2.2.5) [LSB]	msync(GLIBC_2.2.5) [SUSv4]	munlock(GLIBC_2.2.5) [SUSv4]
munlockall(GLIBC_2.2.5) [SUSv4]	munmap(GLIBC_2.2.5) [SUSv4]	nanosleep(GLIBC_2.2.5) [SUSv4]	nice(GLIBC_2.2.5) [SUSv4]
open(GLIBC_2.2.5) [LSB]	opendir(GLIBC_2.2.5) [LSB]	pathconf(GLIBC_2.2.5) [LSB]	pause(GLIBC_2.2.5) [LSB]

5) [SUSv4]	.2.5) [SUSv4]	2.2.5) [SUSv4]	.5) [SUSv4]
pipe(GLIBC_2.2.5) [SUSv4]	poll(GLIBC_2.2.5) [SUSv4]	pread(GLIBC_2.2.5) [SUSv4]	pselect(GLIBC_2.2.5) [SUSv4]
ptrace(GLIBC_2.2.5) [LSB]	pwrite(GLIBC_2.2.5) [SUSv4]	read(GLIBC_2.2.5) [SUSv4]	readdir(GLIBC_2.2.5) [SUSv4]
readdir_r(GLIBC_2.2.5) [SUSv4]	readlink(GLIBC_2.2.5) [SUSv4]	readv(GLIBC_2.2.5) [SUSv4]	rename(GLIBC_2.2.5) [SUSv4]
rmdir(GLIBC_2.2.5) [SUSv4]	sbrk(GLIBC_2.2.5) [SUSv2]	sched_get_priority_max(GLIBC_2.2.5) [SUSv4]	sched_get_priority_min(GLIBC_2.2.5) [SUSv4]
sched_getparam(GLIBC_2.2.5) [SUSv4]	sched_getschedule_r(GLIBC_2.2.5) [SUSv4]	sched_rr_get_interval(GLIBC_2.2.5) [SUSv4]	sched_setparam(GLIBC_2.2.5) [SUSv4]
sched_setscheduler(GLIBC_2.2.5) [LSB]	sched_yield(GLIBC_2.2.5) [SUSv4]	select(GLIBC_2.2.5) [SUSv4]	setcontext(GLIBC_2.2.5) [SUSv3]
setegid(GLIBC_2.2.5) [SUSv4]	seteuid(GLIBC_2.2.5) [SUSv4]	setgid(GLIBC_2.2.5) [SUSv4]	setitimer(GLIBC_2.2.5) [SUSv4]
setpgid(GLIBC_2.2.5) [SUSv4]	setpgrp(GLIBC_2.2.5) [SUSv4]	setpriority(GLIBC_2.2.5) [SUSv4]	setregid(GLIBC_2.2.5) [SUSv4]
setreuid(GLIBC_2.2.5) [SUSv4]	setrlimit(GLIBC_2.2.5) [LSB]	setrlimit64(GLIBC_2.2.5) [LFS]	setsid(GLIBC_2.2.5) [SUSv4]
setuid(GLIBC_2.2.5) [SUSv4]	sleep(GLIBC_2.2.5) [SUSv4]	statfs(GLIBC_2.2.5) [LSB]	statvfs(GLIBC_2.2.5) [SUSv4]
stime(GLIBC_2.2.5) [LSB]	symlink(GLIBC_2.2.5) [SUSv4]	sync(GLIBC_2.2.5) [SUSv4]	sysconf(GLIBC_2.2.5) [LSB]
sysinfo(GLIBC_2.2.5) [LSB]	time(GLIBC_2.2.5) [SUSv4]	times(GLIBC_2.2.5) [SUSv4]	truncate(GLIBC_2.2.5) [SUSv4]
ulimit(GLIBC_2.2.5) [SUSv4]	umask(GLIBC_2.2.5) [SUSv4]	uname(GLIBC_2.2.5) [SUSv4]	unlink(GLIBC_2.2.5) [LSB]
utime(GLIBC_2.2.5) [SUSv4]	utimes(GLIBC_2.2.5) [SUSv4]	vfork(GLIBC_2.2.5) [SUSv3]	wait(GLIBC_2.2.5) [SUSv4]
wait4(GLIBC_2.2.5) [LSB]	waitid(GLIBC_2.2.5) [SUSv4]	waitpid(GLIBC_2.2.5) [SUSv4]	write(GLIBC_2.2.5) [SUSv4]
writelv(GLIBC_2.2.5) [SUSv4]			

An LSB conforming implementation shall provide the architecture specific deprecated functions for System Calls specified in [Table 10-5](#), with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 10-5 libc - System Calls Deprecated Function Interfaces

fstatfs(GLIBC_2.2.5) [LSB]	getdtablesize(GLIBC_2.2.5) [LSB]	getpagesize(GLIBC_2.2.5) [LSB]	getwd(GLIBC_2.2.5) [SUSv3]
statfs(GLIBC_2.2.5) [LSB]			

10.2.4 Standard I/O

10.2.4.1 Interfaces for Standard I/O

An LSB conforming implementation shall provide the architecture specific functions for Standard I/O specified in [Table 10-6](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-6 libc - Standard I/O Function Interfaces

_IO_feof(GLIBC_2.2.5) [LSB]	_IO_getc(GLIBC_2.2.5) [LSB]	_IO_putc(GLIBC_2.2.5) [LSB]	_IO_puts(GLIBC_2.2.5) [LSB]
__fprintf_chk(GLIBC_2.3.4) [LSB]	__printf_chk(GLIBC_2.3.4) [LSB]	__snprintf_chk(GLIBC_2.3.4) [LSB]	__sprintf_chk(GLIBC_2.3.4) [LSB]
__vfprintf_chk(GLIBC_2.3.4) [LSB]	__vprintf_chk(GLIBC_2.3.4) [LSB]	__vsnprintf_chk(GLIBC_2.3.4) [LSB]	__vsprintf_chk(GLIBC_2.3.4) [LSB]
asprintf(GLIBC_2.2.5) [LSB]	clearerr(GLIBC_2.2.5) [SUSv4]	clearerr_unlocked(GLIBC_2.2.5) [LSB]	ctermid(GLIBC_2.2.5) [SUSv4]
dprintf(GLIBC_2.2.5) [SUSv4]	fclose(GLIBC_2.2.5) [SUSv4]	fdopen(GLIBC_2.2.5) [SUSv4]	feof(GLIBC_2.2.5) [SUSv4]
feof_unlocked(GLIBC_2.2.5) [LSB]	ferror(GLIBC_2.2.5) [SUSv4]	ferror_unlocked(GLIBC_2.2.5) [LSB]	fflush(GLIBC_2.2.5) [SUSv4]
fflush_unlocked(GLIBC_2.2.5) [LSB]	fgetc(GLIBC_2.2.5) [SUSv4]	fgetc_unlocked(GLIBC_2.2.5) [LSB]	fgetpos(GLIBC_2.2.5) [SUSv4]
fgets(GLIBC_2.2.5) [SUSv4]	fgets_unlocked(GLIBC_2.2.5) [LSB]	fgetwc_unlocked(GLIBC_2.2.5) [LSB]	fgetws_unlocked(GLIBC_2.2.5) [LSB]
fileno(GLIBC_2.2.5) [SUSv4]	fileno_unlocked(GLIBC_2.2.5) [LSB]	flockfile(GLIBC_2.2.5) [SUSv4]	fopen(GLIBC_2.2.5) [SUSv4]
fprintf(GLIBC_2.2.5) [SUSv4]	fputc(GLIBC_2.2.5) [SUSv4]	fputc_unlocked(GLIBC_2.2.5) [LSB]	fputs(GLIBC_2.2.5) [SUSv4]
fputs_unlocked(GLIBC_2.2.5) [LSB]	fputwc_unlocked(GLIBC_2.2.5) [LSB]	fputws_unlocked(GLIBC_2.2.5) [LSB]	fread(GLIBC_2.2.5) [SUSv4]
fread_unlocked(GLIBC_2.2.5) [LSB]	freopen(GLIBC_2.2.5) [SUSv4]	fscanf(GLIBC_2.2.5) [LSB]	fseek(GLIBC_2.2.5) [SUSv4]
fseeko(GLIBC_2.2.5) [SUSv4]	fsetpos(GLIBC_2.2.5) [SUSv4]	ftell(GLIBC_2.2.5) [SUSv4]	ftello(GLIBC_2.2.5) [SUSv4]
fwrite(GLIBC_2.2.5) [SUSv4]	fwrite_unlocked(GLIBC_2.2.5) [LSB]	getc(GLIBC_2.2.5) [SUSv4]	getc_unlocked(GLIBC_2.2.5) [SUSv4]
getchar(GLIBC_2.2.5) [SUSv4]	getchar_unlocked(GLIBC_2.2.5) [SUSv4]	getdelim(GLIBC_2.2.5) [SUSv4]	getline(GLIBC_2.2.5) [SUSv4]
getw(GLIBC_2.2.5) [SUSv2]	getwc_unlocked(GLIBC_2.2.5)	getwchar_unlocked(GLIBC_2.2.5)	pclose(GLIBC_2.2.5) [SUSv4]

	[LSB]	[LSB]	
popen(GLIBC_2.2.5) [SUSv4]	printf(GLIBC_2.2.5) [SUSv4]	putc(GLIBC_2.2.5) [SUSv4]	putc_unlocked(GLIBC_2.2.5) [SUSv4]
putchar(GLIBC_2.2.5) [SUSv4]	putchar_unlocked(GLIBC_2.2.5) [SUSv4]	puts(GLIBC_2.2.5) [SUSv4]	putw(GLIBC_2.2.5) [SUSv2]
putwc_unlocked(GLIBC_2.2.5) [LSB]	putwchar_unlocked(GLIBC_2.2.5) [LSB]	remove(GLIBC_2.2.5) [SUSv4]	rewind(GLIBC_2.2.5) [SUSv4]
rewinddir(GLIBC_2.2.5) [SUSv4]	scanf(GLIBC_2.2.5) [LSB]	seekdir(GLIBC_2.2.5) [SUSv4]	setbuf(GLIBC_2.2.5) [SUSv4]
setbuffer(GLIBC_2.2.5) [LSB]	setvbuf(GLIBC_2.2.5) [SUSv4]	snprintf(GLIBC_2.2.5) [SUSv4]	sprintf(GLIBC_2.2.5) [SUSv4]
sscanf(GLIBC_2.2.5) [LSB]	telldir(GLIBC_2.2.5) [SUSv4]	tempnam(GLIBC_2.2.5) [SUSv4]	ungetc(GLIBC_2.2.5) [SUSv4]
vasprintf(GLIBC_2.2.5) [LSB]	vdprintf(GLIBC_2.2.5) [SUSv4]	vfprintf(GLIBC_2.2.5) [SUSv4]	vprintf(GLIBC_2.2.5) [SUSv4]
vsnprintf(GLIBC_2.2.5) [SUSv4]	vsprintf(GLIBC_2.2.5) [SUSv4]		

An LSB conforming implementation shall provide the architecture specific deprecated functions for Standard I/O specified in [Table 10-7](#), with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 10-7 libc - Standard I/O Deprecated Function Interfaces

tempnam(GLIBC_2.2.5) [SUSv4]			
--	--	--	--

An LSB conforming implementation shall provide the architecture specific data interfaces for Standard I/O specified in [Table 10-8](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-8 libc - Standard I/O Data Interfaces

stderr(GLIBC_2.2.5) [SUSv4]	stdin(GLIBC_2.2.5) [SUSv4]	stdout(GLIBC_2.2.5) [SUSv4]	
---	--	---	--

10.2.5 Signal Handling

10.2.5.1 Interfaces for Signal Handling

An LSB conforming implementation shall provide the architecture specific functions for Signal Handling specified in [Table 10-9](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-9 libc - Signal Handling Function Interfaces

__libc_current_sigrtmax(GLIBC_2.2.5) [LSB]	__libc_current_sigrtmin(GLIBC_2.2.5) [LSB]	__sigsetjmp(GLIBC_2.2.5) [LSB]	__sysv_signal(GLIBC_2.2.5) [LSB]
__xpg_sigpause(GLIBC_2.2.5) [LSB]	bsd_signal(GLIBC_2.2.5) [SUSv3]	psignal(GLIBC_2.2.5) [SUSv4]	raise(GLIBC_2.2.5) [SUSv4]

sigaction(GLIBC_2.2.5) [SUSv4]	sigaddset(GLIBC_2.2.5) [SUSv4]	sigaltstack(GLIBC_2.2.5) [SUSv4]	sigandset(GLIBC_2.2.5) [LSB]
sigdelset(GLIBC_2.2.5) [SUSv4]	sigemptyset(GLIBC_2.2.5) [SUSv4]	sigfillset(GLIBC_2.2.5) [SUSv4]	sighold(GLIBC_2.2.5) [SUSv4]
sigignore(GLIBC_2.2.5) [SUSv4]	siginterrupt(GLIBC_2.2.5) [SUSv4]	sigisemptyset(GLIBC_2.2.5) [LSB]	sigismember(GLIBC_2.2.5) [SUSv4]
siglongjmp(GLIBC_2.2.5) [SUSv4]	signal(GLIBC_2.2.5) [SUSv4]	sigorset(GLIBC_2.2.5) [LSB]	sigpause(GLIBC_2.2.5) [LSB]
sigpending(GLIBC_2.2.5) [SUSv4]	sigprocmask(GLIBC_2.2.5) [SUSv4]	sigqueue(GLIBC_2.2.5) [SUSv4]	sigrelse(GLIBC_2.2.5) [SUSv4]
sigreturn(GLIBC_2.2.5) [LSB]	sigset(GLIBC_2.2.5) [SUSv4]	sigsuspend(GLIBC_2.2.5) [SUSv4]	sigtimedwait(GLIBC_2.2.5) [SUSv4]
sigwait(GLIBC_2.2.5) [SUSv4]	sigwaitinfo(GLIBC_2.2.5) [SUSv4]		

An LSB conforming implementation shall provide the architecture specific deprecated functions for Signal Handling specified in [Table 10-10](#), with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 10-10 libc - Signal Handling Deprecated Function Interfaces

sigpause(GLIBC_2.2.5) [LSB]			
-----------------------------	--	--	--

An LSB conforming implementation shall provide the architecture specific data interfaces for Signal Handling specified in [Table 10-11](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-11 libc - Signal Handling Data Interfaces

_sys_siglist(GLIBC_2.3.3) [LSB]			
---------------------------------	--	--	--

10.2.6 Localization Functions

10.2.6.1 Interfaces for Localization Functions

An LSB conforming implementation shall provide the architecture specific functions for Localization Functions specified in [Table 10-12](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-12 libc - Localization Functions Function Interfaces

bind_textdomain_codeset(GLIBC_2.2.5) [LSB]	bindtextdomain(GLIBC_2.2.5) [LSB]	catclose(GLIBC_2.2.5) [SUSv4]	catgets(GLIBC_2.2.5) [SUSv4]
catopen(GLIBC_2.2.5) [SUSv4]	dcgettext(GLIBC_2.2.5) [LSB]	dcngettext(GLIBC_2.2.5) [LSB]	dgettext(GLIBC_2.2.5) [LSB]
dngettext(GLIBC_2.2.5) [LSB]	gettext(GLIBC_2.2.5) [LSB]	iconv(GLIBC_2.2.5) [SUSv4]	iconv_close(GLIBC_2.2.5) [SUSv4]
iconv_open(GLIBC_2.2.5) [SUSv4]	localeconv(GLIBC_2.2.5) [SUSv4]	ngettext(GLIBC_2.2.5) [LSB]	nl_langinfo(GLIBC_2.2.5) [SUSv4]

setlocale(GLIBC_2.2.5) [SUSv4]	textdomain(GLIBC_2.2.5) [LSB]		
--------------------------------	-------------------------------	--	--

An LSB conforming implementation shall provide the architecture specific data interfaces for Localization Functions specified in [Table 10-13](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-13 libc - Localization Functions Data Interfaces

_nl_msg_cat_cntr(GLIBC_2.2.5) [LSB]			
-------------------------------------	--	--	--

10.2.7 Posix Spawn Option

10.2.7.1 Interfaces for Posix Spawn Option

An LSB conforming implementation shall provide the architecture specific functions for Posix Spawn Option specified in [Table 10-14](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-14 libc - Posix Spawn Option Function Interfaces

posix_spawn(GLIBC_2.15) [SUSv4]	posix_spawn_file_actions_addclose(GLIBC_2.2.5) [SUSv4]	posix_spawn_file_actions_adddup2(GLIBC_2.2.5) [SUSv4]	posix_spawn_file_actions_addopen(GLIBC_2.2.5) [SUSv4]
posix_spawn_file_actions_destroy(GLIBC_2.2.5) [SUSv4]	posix_spawn_file_actions_init(GLIBC_2.2.5) [SUSv4]	posix_spawnattr_destroy(GLIBC_2.2.5) [SUSv4]	posix_spawnattr_getflags(GLIBC_2.2.5) [SUSv4]
posix_spawnattr_getpgroup(GLIBC_2.2.5) [SUSv4]	posix_spawnattr_getschedparam(GLIBC_2.2.5) [SUSv4]	posix_spawnattr_getschedpolicy(GLIBC_2.2.5) [SUSv4]	posix_spawnattr_getsigdefault(GLIBC_2.2.5) [SUSv4]
posix_spawnattr_getsigmask(GLIBC_2.2.5) [SUSv4]	posix_spawnattr_init(GLIBC_2.2.5) [SUSv4]	posix_spawnattr_setflags(GLIBC_2.2.5) [SUSv4]	posix_spawnattr_setpgroup(GLIBC_2.2.5) [SUSv4]
posix_spawnattr_setschedparam(GLIBC_2.2.5) [SUSv4]	posix_spawnattr_setschedpolicy(GLIBC_2.2.5) [SUSv4]	posix_spawnattr_setsigdefault(GLIBC_2.2.5) [SUSv4]	posix_spawnattr_setsigmask(GLIBC_2.2.5) [SUSv4]
posix_spawnnp(GLIBC_2.15) [SUSv4]			

10.2.8 Posix Advisory Option

10.2.8.1 Interfaces for Posix Advisory Option

An LSB conforming implementation shall provide the architecture specific functions for Posix Advisory Option specified in [Table 10-15](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-15 libc - Posix Advisory Option Function Interfaces

posix_fadvise(GLIBC_2.2.5) [SUSv4]	posix_fallocate(GLIBC_2.2.5) [SUSv4]	posix_madvise(GLIBC_2.2.5) [SUSv4]	posix_memalign(GLIBC_2.2.5) [SUSv4]
------------------------------------	--------------------------------------	------------------------------------	-------------------------------------

10.2.9 Socket Interface

10.2.9.1 Interfaces for Socket Interface

An LSB conforming implementation shall provide the architecture specific functions for Socket Interface specified in [Table 10-16](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-16 libc - Socket Interface Function Interfaces

__h_errno_location(GLIBC_2.2.5) [LSB]	accept(GLIBC_2.2.5) [SUSv4]	bind(GLIBC_2.2.5) [SUSv4]	bindresvport(GLIBC_2.2.5) [LSB]
connect(GLIBC_2.2.5) [SUSv4]	gethostid(GLIBC_2.2.5) [SUSv4]	gethostname(GLIBC_2.2.5) [SUSv4]	getpeername(GLIBC_2.2.5) [SUSv4]
getsockname(GLIBC_2.2.5) [SUSv4]	getsockopt(GLIBC_2.2.5) [LSB]	if_freenameindex(GLIBC_2.2.5) [SUSv4]	if_indextoname(GLIBC_2.2.5) [SUSv4]
if_nameindex(GLIBC_2.2.5) [SUSv4]	if_nametoindex(GLIBC_2.2.5) [SUSv4]	listen(GLIBC_2.2.5) [SUSv4]	recv(GLIBC_2.2.5) [SUSv4]
recvfrom(GLIBC_2.2.5) [SUSv4]	recvmsg(GLIBC_2.2.5) [SUSv4]	send(GLIBC_2.2.5) [SUSv4]	sendmsg(GLIBC_2.2.5) [SUSv4]
sendto(GLIBC_2.2.5) [SUSv4]	setsockopt(GLIBC_2.2.5) [LSB]	shutdown(GLIBC_2.2.5) [SUSv4]	socketmark(GLIBC_2.2.5) [SUSv4]
socket(GLIBC_2.2.5) [SUSv4]	socketpair(GLIBC_2.2.5) [SUSv4]		

An LSB conforming implementation shall provide the architecture specific data interfaces for Socket Interface specified in [Table 10-17](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-17 libc - Socket Interface Data Interfaces

in6addr_any(GLIBC_2.2.5) [SUSv3]	in6addr_loopback(GLIBC_2.2.5) [SUSv3]		
--	---	--	--

10.2.10 Wide Characters

10.2.10.1 Interfaces for Wide Characters

An LSB conforming implementation shall provide the architecture specific functions for Wide Characters specified in [Table 10-18](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-18 libc - Wide Characters Function Interfaces

__wcstod_internal(GLIBC_2.2.5) [LSB]	__wcstof_internal(GLIBC_2.2.5) [LSB]	__wcstol_internal(GLIBC_2.2.5) [LSB]	__wcstold_internal(GLIBC_2.2.5) [LSB]
__wcstoul_internal(GLIBC_2.2.5) [LSB]	btowc(GLIBC_2.2.5) [SUSv4]	fgetwc(GLIBC_2.2.5) [SUSv4]	fgetws(GLIBC_2.2.5) [SUSv4]
fputwc(GLIBC_2.2.5) [SUSv4]	fputws(GLIBC_2.2.5) [SUSv4]	fwide(GLIBC_2.2.5) [SUSv4]	fwprintf(GLIBC_2.2.5) [SUSv4]
fwscanf(GLIBC_2.2.5) [SUSv4]	getwc(GLIBC_2.2.5) [SUSv4]	getwchar(GLIBC_2.2.5) [SUSv4]	mblen(GLIBC_2.2.5) [SUSv4]

.2.5) [LSB]	.5) [SUSv4]	2.2.5) [SUSv4]	.5) [SUSv4]
mbrlen(GLIBC_2.2.5) [SUSv4]	mbrtowc(GLIBC_2.2.5) [SUSv4]	mbsinit(GLIBC_2.2.5) [SUSv4]	mbsnrtowcs(GLIBC_2.2.5) [SUSv4]
mbsrtowcs(GLIBC_2.2.5) [SUSv4]	mbstowcs(GLIBC_2.2.5) [SUSv4]	mbtowc(GLIBC_2.2.5) [SUSv4]	putwc(GLIBC_2.2.5) [SUSv4]
putwchar(GLIBC_2.2.5) [SUSv4]	swprintf(GLIBC_2.2.5) [SUSv4]	swscanf(GLIBC_2.2.5) [LSB]	towctrans(GLIBC_2.2.5) [SUSv4]
towlower(GLIBC_2.2.5) [SUSv4]	towupper(GLIBC_2.2.5) [SUSv4]	ungetwc(GLIBC_2.2.5) [SUSv4]	vfwprintf(GLIBC_2.2.5) [SUSv4]
vfwscanf(GLIBC_2.2.5) [LSB]	vswprintf(GLIBC_2.2.5) [SUSv4]	vswscanf(GLIBC_2.2.5) [LSB]	vwprintf(GLIBC_2.2.5) [SUSv4]
vwscanf(GLIBC_2.2.5) [LSB]	wcpcpy(GLIBC_2.2.5) [SUSv4]	wcpncpy(GLIBC_2.2.5) [SUSv4]	wcrtomb(GLIBC_2.2.5) [SUSv4]
wscasecmp(GLIBC_2.2.5) [SUSv4]	wscat(GLIBC_2.2.5) [SUSv4]	wcschr(GLIBC_2.2.5) [SUSv4]	wscmp(GLIBC_2.2.5) [SUSv4]
wscoll(GLIBC_2.2.5) [SUSv4]	wscopy(GLIBC_2.2.5) [SUSv4]	wscspn(GLIBC_2.2.5) [SUSv4]	wcsdup(GLIBC_2.2.5) [SUSv4]
wcsftime(GLIBC_2.2.5) [SUSv4]	wcslen(GLIBC_2.2.5) [SUSv4]	wcsncasecmp(GLIBC_2.2.5) [SUSv4]	wcsncat(GLIBC_2.2.5) [SUSv4]
wcsncmp(GLIBC_2.2.5) [SUSv4]	wcsncpy(GLIBC_2.2.5) [SUSv4]	wcsnlen(GLIBC_2.2.5) [SUSv4]	wcsnrtombs(GLIBC_2.2.5) [SUSv4]
wcsprk(GLIBC_2.2.5) [SUSv4]	wcsrchr(GLIBC_2.2.5) [SUSv4]	wcsrtombs(GLIBC_2.2.5) [SUSv4]	wcsspn(GLIBC_2.2.5) [SUSv4]
wcsstr(GLIBC_2.2.5) [SUSv4]	wctod(GLIBC_2.2.5) [SUSv4]	wctof(GLIBC_2.2.5) [SUSv4]	wcstoimax(GLIBC_2.2.5) [SUSv4]
wctok(GLIBC_2.2.5) [SUSv4]	wctol(GLIBC_2.2.5) [SUSv4]	wctold(GLIBC_2.2.5) [SUSv4]	wctoll(GLIBC_2.2.5) [SUSv4]
wctombs(GLIBC_2.2.5) [SUSv4]	wctoq(GLIBC_2.2.5) [LSB]	wctoul(GLIBC_2.2.5) [SUSv4]	wctoull(GLIBC_2.2.5) [SUSv4]
wctoumax(GLIBC_2.2.5) [SUSv4]	wctouq(GLIBC_2.2.5) [LSB]	wcswcs(GLIBC_2.2.5) [SUSv3]	wcswidth(GLIBC_2.2.5) [SUSv4]
wcsxfrm(GLIBC_2.2.5) [SUSv4]	wctob(GLIBC_2.2.5) [SUSv4]	wctomb(GLIBC_2.2.5) [SUSv4]	wctrans(GLIBC_2.2.5) [SUSv4]
wctype(GLIBC_2.2.5) [SUSv4]	wcwidth(GLIBC_2.2.5) [SUSv4]	wmemchr(GLIBC_2.2.5) [SUSv4]	wmemcmp(GLIBC_2.2.5) [SUSv4]
wmemcpy(GLIBC_2.2.5) [SUSv4]	wmemmove(GLIBC_2.2.5) [SUSv4]	wmemset(GLIBC_2.2.5) [SUSv4]	wprintf(GLIBC_2.2.5) [SUSv4]
wscanf(GLIBC_2.2.5) [LSB]			

10.2.11 String Functions

10.2.11.1 Interfaces for String Functions

An LSB conforming implementation shall provide the architecture specific functions for String Functions specified in [Table 10-19](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-19 libc - String Functions Function Interfaces

<code>__memcpy(GLIBC_2.2.5)</code> [LSB]	<code>__rawmemchr(GLIBC_2.2.5)</code> [LSB]	<code>__stpcpy(GLIBC_2.2.5)</code> [LSB]	<code>__strdup(GLIBC_2.2.5)</code> [LSB]
<code>__strtod_internal(GLIBC_2.2.5)</code> [LSB]	<code>__strtof_internal(GLIBC_2.2.5)</code> [LSB]	<code>__strtok_r(GLIBC_2.2.5)</code> [LSB]	<code>__strtol_internal(GLIBC_2.2.5)</code> [LSB]
<code>__strtold_internal(GLIBC_2.2.5)</code> [LSB]	<code>__strtoll_internal(GLIBC_2.2.5)</code> [LSB]	<code>__strtoul_internal(GLIBC_2.2.5)</code> [LSB]	<code>__strtoull_internal(GLIBC_2.2.5)</code> [LSB]
<code>__xpg_strerror_r(GLIBC_2.3.4)</code> [LSB]	<code>bcmp(GLIBC_2.2.5)</code> [SUSv3]	<code>bcopy(GLIBC_2.2.5)</code> [SUSv3]	<code>bzero(GLIBC_2.2.5)</code> [SUSv3]
<code>ffs(GLIBC_2.2.5)</code> [SUSv4]	<code>index(GLIBC_2.2.5)</code> [SUSv3]	<code>memcpy(GLIBC_2.2.5)</code> [SUSv4]	<code>memchr(GLIBC_2.2.5)</code> [SUSv4]
<code>memcmp(GLIBC_2.2.5)</code> [SUSv4]	<code>memcpy(GLIBC_2.14)</code> [SUSv4]	<code>memmove(GLIBC_2.2.5)</code> [SUSv4]	<code>memrchr(GLIBC_2.2.5)</code> [LSB]
<code>memset(GLIBC_2.2.5)</code> [SUSv4]	<code>rindex(GLIBC_2.2.5)</code> [SUSv3]	<code>stpcpy(GLIBC_2.2.5)</code> [SUSv4]	<code>stpncpy(GLIBC_2.2.5)</code> [SUSv4]
<code>strcasemp(GLIBC_2.2.5)</code> [SUSv4]	<code>strcasestr(GLIBC_2.2.5)</code> [LSB]	<code>strcat(GLIBC_2.2.5)</code> [SUSv4]	<code>strchr(GLIBC_2.2.5)</code> [SUSv4]
<code>strcmp(GLIBC_2.2.5)</code> [SUSv4]	<code>strcoll(GLIBC_2.2.5)</code> [SUSv4]	<code>strcpy(GLIBC_2.2.5)</code> [SUSv4]	<code>strcspn(GLIBC_2.2.5)</code> [SUSv4]
<code>strdup(GLIBC_2.2.5)</code> [SUSv4]	<code>strerror(GLIBC_2.2.5)</code> [SUSv4]	<code>strerror_r(GLIBC_2.2.5)</code> [LSB]	<code>strfmon(GLIBC_2.2.5)</code> [SUSv4]
<code>strftime(GLIBC_2.2.5)</code> [SUSv4]	<code>strlen(GLIBC_2.2.5)</code> [SUSv4]	<code>strncasemp(GLIBC_2.2.5)</code> [SUSv4]	<code>strncat(GLIBC_2.2.5)</code> [SUSv4]
<code>strncmp(GLIBC_2.2.5)</code> [SUSv4]	<code>strncpy(GLIBC_2.2.5)</code> [SUSv4]	<code>strndup(GLIBC_2.2.5)</code> [SUSv4]	<code>strnlen(GLIBC_2.2.5)</code> [SUSv4]
<code>strpbrk(GLIBC_2.2.5)</code> [SUSv4]	<code>strptime(GLIBC_2.2.5)</code> [LSB]	<code>strchr(GLIBC_2.2.5)</code> [SUSv4]	<code>strsep(GLIBC_2.2.5)</code> [LSB]
<code>strsignal(GLIBC_2.2.5)</code> [SUSv4]	<code>strspn(GLIBC_2.2.5)</code> [SUSv4]	<code>strstr(GLIBC_2.2.5)</code> [SUSv4]	<code>strtof(GLIBC_2.2.5)</code> [SUSv4]
<code>strtoimax(GLIBC_2.2.5)</code> [SUSv4]	<code>strtok(GLIBC_2.2.5)</code> [SUSv4]	<code>strtok_r(GLIBC_2.2.5)</code> [SUSv4]	<code>strtold(GLIBC_2.2.5)</code> [SUSv4]
<code>strtoll(GLIBC_2.2.5)</code> [SUSv4]	<code>strtoq(GLIBC_2.2.5)</code> [LSB]	<code>strtoull(GLIBC_2.2.5)</code> [SUSv4]	<code>strtoumax(GLIBC_2.2.5)</code> [SUSv4]
<code>strtouq(GLIBC_2.2.5)</code> [LSB]	<code>strxfrm(GLIBC_2.2.5)</code> [SUSv4]	<code>swab(GLIBC_2.2.5)</code> [SUSv4]	

An LSB conforming implementation shall provide the architecture specific deprecated functions for String Functions specified in [Table 10-20](#), with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 10-20 libc - String Functions Deprecated Function Interfaces

<code>strerror_r(GLIBC_2.2.5)</code> [LSB]			
--	--	--	--

10.2.12 IPC Functions

10.2.12.1 Interfaces for IPC Functions

An LSB conforming implementation shall provide the architecture specific functions for IPC Functions specified in [Table 10-21](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-21 libc - IPC Functions Function Interfaces

ftok(GLIBC_2.2.5) [SUSv4]	msgctl(GLIBC_2.2.5) [SUSv4]	msgget(GLIBC_2.2.5) [SUSv4]	msgrcv(GLIBC_2.2.5) [SUSv4]
msgsnd(GLIBC_2.2.5) [SUSv4]	semctl(GLIBC_2.2.5) [SUSv4]	semget(GLIBC_2.2.5) [SUSv4]	semop(GLIBC_2.2.5) [SUSv4]
shmat(GLIBC_2.2.5) [SUSv4]	shmctl(GLIBC_2.2.5) [SUSv4]	shmdt(GLIBC_2.2.5) [SUSv4]	shmget(GLIBC_2.2.5) [SUSv4]

10.2.13 Regular Expressions

10.2.13.1 Interfaces for Regular Expressions

An LSB conforming implementation shall provide the architecture specific functions for Regular Expressions specified in [Table 10-22](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-22 libc - Regular Expressions Function Interfaces

regcomp(GLIBC_2.2.5) [SUSv4]	regerror(GLIBC_2.2.5) [SUSv4]	regexexec(GLIBC_2.3.4) [LSB]	regfree(GLIBC_2.2.5) [SUSv4]
------------------------------	-------------------------------	------------------------------	------------------------------

10.2.14 Character Type Functions

10.2.14.1 Interfaces for Character Type Functions

An LSB conforming implementation shall provide the architecture specific functions for Character Type Functions specified in [Table 10-23](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-23 libc - Character Type Functions Function Interfaces

__ctype_get_mb_cur_max(GLIBC_2.2.5) [LSB]	_tolower(GLIBC_2.2.5) [SUSv4]	_toupper(GLIBC_2.2.5) [SUSv4]	isalnum(GLIBC_2.2.5) [SUSv4]
isalpha(GLIBC_2.2.5) [SUSv4]	isascii(GLIBC_2.2.5) [SUSv4]	iscntrl(GLIBC_2.2.5) [SUSv4]	isdigit(GLIBC_2.2.5) [SUSv4]
isgraph(GLIBC_2.2.5) [SUSv4]	islower(GLIBC_2.2.5) [SUSv4]	isprint(GLIBC_2.2.5) [SUSv4]	ispunct(GLIBC_2.2.5) [SUSv4]
isspace(GLIBC_2.2.5) [SUSv4]	isupper(GLIBC_2.2.5) [SUSv4]	iswalnum(GLIBC_2.2.5) [SUSv4]	iswalpha(GLIBC_2.2.5) [SUSv4]
iswblank(GLIBC_2.2.5) [SUSv4]	iswcntrl(GLIBC_2.2.5) [SUSv4]	iswctype(GLIBC_2.2.5) [SUSv4]	iswdigit(GLIBC_2.2.5) [SUSv4]
iswgraph(GLIBC_2.2.5) [SUSv4]	iswlower(GLIBC_2.2.5) [SUSv4]	iswprint(GLIBC_2.2.5) [SUSv4]	iswpunct(GLIBC_2.2.5) [SUSv4]
iswspace(GLIBC_2.2.5) [SUSv4]	iswupper(GLIBC_2.2.5) [SUSv4]	iswxdigit(GLIBC_2.2.5) [SUSv4]	isxdigit(GLIBC_2.2.5) [SUSv4]
toascii(GLIBC_2.2.5) [SUSv4]	tolower(GLIBC_2.2.5) [SUSv4]	toupper(GLIBC_2.2.5) [SUSv4]	

10.2.15 Time Manipulation

10.2.15.1 Interfaces for Time Manipulation

An LSB conforming implementation shall provide the architecture specific functions for Time Manipulation specified in [Table 10-24](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-24 libc - Time Manipulation Function Interfaces

adjtime(GLIBC_2.2.5) [LSB]	asctime(GLIBC_2.2.5) [SUSv4]	asctime_r(GLIBC_2.2.5) [SUSv4]	ctime(GLIBC_2.2.5) [SUSv4]
ctime_r(GLIBC_2.2.5) [SUSv4]	difftime(GLIBC_2.2.5) [SUSv4]	gmtime(GLIBC_2.2.5) [SUSv4]	gmtime_r(GLIBC_2.2.5) [SUSv4]
localtime(GLIBC_2.2.5) [SUSv4]	localtime_r(GLIBC_2.2.5) [SUSv4]	mktime(GLIBC_2.2.5) [SUSv4]	tzset(GLIBC_2.2.5) [SUSv4]
ualarm(GLIBC_2.2.5) [SUSv3]			

An LSB conforming implementation shall provide the architecture specific data interfaces for Time Manipulation specified in [Table 10-25](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-25 libc - Time Manipulation Data Interfaces

__daylight(GLIBC_2.2.5) [LSB]	__timezone(GLIBC_2.2.5) [LSB]	__tzname(GLIBC_2.2.5) [LSB]	daylight(GLIBC_2.2.5) [SUSv4]
timezone(GLIBC_2.2.5) [SUSv4]	tzname(GLIBC_2.2.5) [SUSv4]		

10.2.16 Terminal Interface Functions

10.2.16.1 Interfaces for Terminal Interface Functions

An LSB conforming implementation shall provide the architecture specific functions for Terminal Interface Functions specified in [Table 10-26](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-26 libc - Terminal Interface Functions Function Interfaces

cfgetispeed(GLIBC_2.2.5) [SUSv4]	cfgetospeed(GLIBC_2.2.5) [SUSv4]	cfmakeraw(GLIBC_2.2.5) [LSB]	cfsetispeed(GLIBC_2.2.5) [SUSv4]
cfsetospeed(GLIBC_2.2.5) [SUSv4]	cfsetspeed(GLIBC_2.2.5) [LSB]	tcdrain(GLIBC_2.2.5) [SUSv4]	tcflow(GLIBC_2.2.5) [SUSv4]
tcflush(GLIBC_2.2.5) [SUSv4]	tcgetattr(GLIBC_2.2.5) [SUSv4]	tcgetpgrp(GLIBC_2.2.5) [SUSv4]	tcgetsid(GLIBC_2.2.5) [SUSv4]
tcsendbreak(GLIBC_2.2.5) [SUSv4]	tcsetattr(GLIBC_2.2.5) [SUSv4]	tcsetpgrp(GLIBC_2.2.5) [SUSv4]	

10.2.17 System Database Interface

10.2.17.1 Interfaces for System Database Interface

An LSB conforming implementation shall provide the architecture specific functions for System Database Interface specified in [Table 10-27](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-27 libc - System Database Interface Function Interfaces

endgrent(GLIBC_2.2.5) [SUSv4]	endprotoent(GLIBC_2.2.5) [SUSv4]	endpwent(GLIBC_2.2.5) [SUSv4]	endservent(GLIBC_2.2.5) [SUSv4]
---	--	---	---

2.2.5) [SUSv4]	C_2.2.5) [SUSv4]	_2.2.5) [SUSv4]	C_2.2.5) [SUSv4]
endutent(GLIBC_2.2.5) [LSB]	endutxent(GLIBC_2.2.5) [SUSv4]	getgrent(GLIBC_2.2.5) [SUSv4]	getgrent_r(GLIBC_2.2.5) [LSB]
getgrgid(GLIBC_2.2.5) [SUSv4]	getgrgid_r(GLIBC_2.2.5) [SUSv4]	getgrnam(GLIBC_2.2.5) [SUSv4]	getgrnam_r(GLIBC_2.2.5) [SUSv4]
getgrouplist(GLIBC_2.2.5) [LSB]	gethostbyaddr(GLIBC_2.2.5) [SUSv3]	gethostbyaddr_r(GLIBC_2.2.5) [LSB]	gethostbyname(GLIBC_2.2.5) [SUSv3]
gethostbyname2(GLIBC_2.2.5) [LSB]	gethostbyname2_r(GLIBC_2.2.5) [LSB]	gethostbyname_r(GLIBC_2.2.5) [LSB]	getprotobyname(GLIBC_2.2.5) [SUSv4]
getprotobyname_r(GLIBC_2.2.5) [LSB]	getprotobynumber(GLIBC_2.2.5) [SUSv4]	getprotobynumber_r(GLIBC_2.2.5) [LSB]	getprotoent(GLIBC_2.2.5) [SUSv4]
getprotoent_r(GLIBC_2.2.5) [LSB]	getpwent(GLIBC_2.2.5) [SUSv4]	getpwent_r(GLIBC_2.2.5) [LSB]	getpwnam(GLIBC_2.2.5) [SUSv4]
getpwnam_r(GLIBC_2.2.5) [SUSv4]	getpwuid(GLIBC_2.2.5) [SUSv4]	getpwuid_r(GLIBC_2.2.5) [SUSv4]	getservbyname(GLIBC_2.2.5) [SUSv4]
getservbyname_r(GLIBC_2.2.5) [LSB]	getservbyport(GLIBC_2.2.5) [SUSv4]	getservbyport_r(GLIBC_2.2.5) [LSB]	getservent(GLIBC_2.2.5) [SUSv4]
getservent_r(GLIBC_2.2.5) [LSB]	getutent(GLIBC_2.2.5) [LSB]	getutent_r(GLIBC_2.2.5) [LSB]	getutxent(GLIBC_2.2.5) [SUSv4]
getutxid(GLIBC_2.2.5) [SUSv4]	getutxline(GLIBC_2.2.5) [SUSv4]	pututxline(GLIBC_2.2.5) [SUSv4]	setgrent(GLIBC_2.2.5) [SUSv4]
setgroups(GLIBC_2.2.5) [LSB]	setprotoent(GLIBC_2.2.5) [SUSv4]	setpwent(GLIBC_2.2.5) [SUSv4]	setservent(GLIBC_2.2.5) [SUSv4]
setutent(GLIBC_2.2.5) [LSB]	setutxent(GLIBC_2.2.5) [SUSv4]	utmpname(GLIBC_2.2.5) [LSB]	

An LSB conforming implementation shall provide the architecture specific deprecated functions for System Database Interface specified in [Table 10-28](#), with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 10-28 libc - System Database Interface Deprecated Function Interfaces

gethostbyaddr(GLIBC_2.2.5) [SUSv3]	gethostbyaddr_r(GLIBC_2.2.5) [LSB]	gethostbyname(GLIBC_2.2.5) [SUSv3]	gethostbyname2(GLIBC_2.2.5) [LSB]
gethostbyname2_r(GLIBC_2.2.5) [LSB]	gethostbyname_r(GLIBC_2.2.5) [LSB]		

10.2.18 Language Support

10.2.18.1 Interfaces for Language Support

An LSB conforming implementation shall provide the architecture specific functions for Language Support specified in [Table 10-29](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-29 libc - Language Support Function Interfaces

<code>__libc_start_main(</code> GLIBC_2.2.5) [LSB]			
--	--	--	--

10.2.19 Large File Support

10.2.19.1 Interfaces for Large File Support

An LSB conforming implementation shall provide the architecture specific functions for Large File Support specified in [Table 10-30](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-30 libc - Large File Support Function Interfaces

<code>__fxstat64(</code> GLIBC_2.2.5) [LSB]	<code>__lxstat64(</code> GLIBC_2.2.5) [LSB]	<code>__xstat64(</code> GLIBC_2.2.5) [LSB]	<code>creat64(</code> GLIBC_2.2.5) [LFS]
<code>fgetpos64(</code> GLIBC_2.2.5) [LFS]	<code>fopen64(</code> GLIBC_2.2.5) [LFS]	<code>freopen64(</code> GLIBC_2.2.5) [LFS]	<code>fseeko64(</code> GLIBC_2.2.5) [LFS]
<code>fsetpos64(</code> GLIBC_2.2.5) [LFS]	<code>fstatfs64(</code> GLIBC_2.2.5) [LSB]	<code>fstatvfs64(</code> GLIBC_2.2.5) [LFS]	<code>ftello64(</code> GLIBC_2.2.5) [LFS]
<code>ftruncate64(</code> GLIBC_2.2.5) [LFS]	<code>ftw64(</code> GLIBC_2.2.5) [LFS]	<code>getrlimit64(</code> GLIBC_2.2.5) [LFS]	<code>lockf64(</code> GLIBC_2.2.5) [LFS]
<code>lseek64(</code> GLIBC_2.2.5) [LFS]	<code>mkstemp64(</code> GLIBC_2.2.5) [LSB]	<code>mmap64(</code> GLIBC_2.2.5) [LFS]	<code>nftw64(</code> GLIBC_2.2.5) [LFS]
<code>open64(</code> GLIBC_2.2.5) [LFS]	<code>posix_fadvise64(</code> GLIBC_2.2.5) [LSB]	<code>posix_fallocate64(</code> GLIBC_2.2.5) [LSB]	<code>pread64(</code> GLIBC_2.2.5) [LSB]
<code>pwrite64(</code> GLIBC_2.2.5) [LSB]	<code>readdir64(</code> GLIBC_2.2.5) [LFS]	<code>readdir64_r(</code> GLIBC_2.2.5) [LSB]	<code>statfs64(</code> GLIBC_2.2.5) [LSB]
<code>statvfs64(</code> GLIBC_2.2.5) [LFS]	<code>tmpfile64(</code> GLIBC_2.2.5) [LFS]	<code>truncate64(</code> GLIBC_2.2.5) [LFS]	

An LSB conforming implementation shall provide the architecture specific deprecated functions for Large File Support specified in [Table 10-31](#), with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 10-31 libc - Large File Support Deprecated Function Interfaces

<code>fstatfs64(</code> GLIBC_2.2.5) [LSB]	<code>statfs64(</code> GLIBC_2.2.5) [LSB]		
---	--	--	--

10.2.20 Inotify

10.2.20.1 Interfaces for Inotify

No external functions are defined for libc - Inotify in this part of the specification. See also the generic specification.

10.2.21 Standard Library

10.2.21.1 Interfaces for Standard Library

An LSB conforming implementation shall provide the architecture specific functions for Standard Library specified in [Table 10-32](#), with the full mandatory functionality as de-

scribed in the referenced underlying specification.

Table 10-32 libc - Standard Library Function Interfaces

_Exit(GLIBC_2.2.5) [SUSv4]	__assert_fail(GLIBC_2.2.5) [LSB]	__cxa_atexit(GLIBC_2.2.5) [LSB]	__cxa_finalize(GLIBC_2.2.5) [LSB]
__errno_location(GLIBC_2.2.5) [LSB]	__fpending(GLIBC_2.2.5) [LSB]	__getpagesize(GLIBC_2.2.5) [LSB]	__isinf(GLIBC_2.2.5) [LSB]
__isinff(GLIBC_2.2.5) [LSB]	__isinfl(GLIBC_2.2.5) [LSB]	__isnan(GLIBC_2.2.5) [LSB]	__isnanf(GLIBC_2.2.5) [LSB]
__isnani(GLIBC_2.2.5) [LSB]	__sysconf(GLIBC_2.2.5) [LSB]	__xpg_basename(GLIBC_2.2.5) [LSB]	_exit(GLIBC_2.2.5) [SUSv4]
_longjmp(GLIBC_2.2.5) [SUSv4]	_setjmp(GLIBC_2.2.5) [SUSv4]	a64l(GLIBC_2.2.5) [SUSv4]	abort(GLIBC_2.2.5) [SUSv4]
abs(GLIBC_2.2.5) [SUSv4]	alphasort(GLIBC_2.2.5) [SUSv4]	alphasort64(GLIBC_2.2.5) [LSB]	argz_add(GLIBC_2.2.5) [LSB]
argz_add_sep(GLIBC_2.2.5) [LSB]	argz_append(GLIBC_2.2.5) [LSB]	argz_count(GLIBC_2.2.5) [LSB]	argz_create(GLIBC_2.2.5) [LSB]
argz_create_sep(GLIBC_2.2.5) [LSB]	argz_delete(GLIBC_2.2.5) [LSB]	argz_extract(GLIBC_2.2.5) [LSB]	argz_insert(GLIBC_2.2.5) [LSB]
argz_next(GLIBC_2.2.5) [LSB]	argz_replace(GLIBC_2.2.5) [LSB]	argz_stringify(GLIBC_2.2.5) [LSB]	atof(GLIBC_2.2.5) [SUSv4]
atoi(GLIBC_2.2.5) [SUSv4]	atol(GLIBC_2.2.5) [SUSv4]	atoll(GLIBC_2.2.5) [SUSv4]	basename(GLIBC_2.2.5) [LSB]
bsearch(GLIBC_2.2.5) [SUSv4]	calloc(GLIBC_2.2.5) [SUSv4]	closelog(GLIBC_2.2.5) [SUSv4]	confstr(GLIBC_2.2.5) [SUSv4]
cuserid(GLIBC_2.2.5) [SUSv2]	daemon(GLIBC_2.2.5) [LSB]	dirfd(GLIBC_2.2.5) [SUSv4]	dirname(GLIBC_2.2.5) [SUSv4]
div(GLIBC_2.2.5) [SUSv4]	dl_iterate_phdr(GLIBC_2.2.5) [LSB]	drand48(GLIBC_2.2.5) [SUSv4]	drand48_r(GLIBC_2.2.5) [LSB]
ecvt(GLIBC_2.2.5) [SUSv3]	envz_add(GLIBC_2.2.5) [LSB]	envz_entry(GLIBC_2.2.5) [LSB]	envz_get(GLIBC_2.2.5) [LSB]
envz_merge(GLIBC_2.2.5) [LSB]	envz_remove(GLIBC_2.2.5) [LSB]	envz_strip(GLIBC_2.2.5) [LSB]	erand48(GLIBC_2.2.5) [SUSv4]
erand48_r(GLIBC_2.2.5) [LSB]	err(GLIBC_2.2.5) [LSB]	error(GLIBC_2.2.5) [LSB]	errx(GLIBC_2.2.5) [LSB]
fcvt(GLIBC_2.2.5) [SUSv3]	fmemopen(GLIBC_2.2.5) [SUSv4]	fmtmsg(GLIBC_2.2.5) [SUSv4]	fnmatch(GLIBC_2.2.5) [LSB]
fpathconf(GLIBC_2.2.5) [SUSv4]	free(GLIBC_2.2.5) [SUSv4]	freeaddrinfo(GLIBC_2.2.5) [SUSv4]	ftrylockfile(GLIBC_2.2.5) [SUSv4]
ftw(GLIBC_2.2.5) [SUSv4]	funlockfile(GLIBC_2.2.5) [SUSv4]	gai_strerror(GLIBC_2.2.5) [SUSv4]	gcvt(GLIBC_2.2.5) [SUSv3]
getaddrinfo(GLIBC_2.2.5) [SUSv4]	getcwd(GLIBC_2.2.5) [LSB]	getdate(GLIBC_2.2.5) [SUSv4]	getdomainname(GLIBC_2.2.5) [LSB]
getenv(GLIBC_2.2.5) [SUSv4]	getlogin(GLIBC_2.2.5) [SUSv4]	getlogin_r(GLIBC_2.2.5) [SUSv4]	getnameinfo(GLIBC_2.2.5) [SUSv4]

LSB Core - X86-64 5.0

			[SUSv4]
getopt(GLIBC_2.2.5) [LSB]	getopt_long(GLIBC_2.2.5) [LSB]	getopt_long_only(GLIBC_2.2.5) [LSB]	getsubopt(GLIBC_2.2.5) [SUSv4]
gettimeofday(GLIBC_2.2.5) [SUSv4]	glob(GLIBC_2.2.5) [SUSv4]	glob64(GLIBC_2.2.5) [LSB]	globfree(GLIBC_2.2.5) [SUSv4]
globfree64(GLIBC_2.2.5) [LSB]	grantpt(GLIBC_2.2.5) [SUSv4]	hcreate(GLIBC_2.2.5) [SUSv4]	hcreate_r(GLIBC_2.2.5) [LSB]
hdestroy(GLIBC_2.2.5) [SUSv4]	hdestroy_r(GLIBC_2.2.5) [LSB]	hsearch(GLIBC_2.2.5) [SUSv4]	hsearch_r(GLIBC_2.2.5) [LSB]
htonl(GLIBC_2.2.5) [SUSv4]	htons(GLIBC_2.2.5) [SUSv4]	imaxabs(GLIBC_2.2.5) [SUSv4]	imaxdiv(GLIBC_2.2.5) [SUSv4]
inet_addr(GLIBC_2.2.5) [SUSv4]	inet_aton(GLIBC_2.2.5) [LSB]	inet_ntoa(GLIBC_2.2.5) [SUSv4]	inet_ntop(GLIBC_2.2.5) [SUSv4]
inet_pton(GLIBC_2.2.5) [SUSv4]	initstate(GLIBC_2.2.5) [SUSv4]	initstate_r(GLIBC_2.2.5) [LSB]	insque(GLIBC_2.2.5) [SUSv4]
isatty(GLIBC_2.2.5) [SUSv4]	isblank(GLIBC_2.2.5) [SUSv4]	jrand48(GLIBC_2.2.5) [SUSv4]	jrand48_r(GLIBC_2.2.5) [LSB]
l64a(GLIBC_2.2.5) [SUSv4]	labs(GLIBC_2.2.5) [SUSv4]	lcong48(GLIBC_2.2.5) [SUSv4]	lcong48_r(GLIBC_2.2.5) [LSB]
ldiv(GLIBC_2.2.5) [SUSv4]	lfind(GLIBC_2.2.5) [SUSv4]	llabs(GLIBC_2.2.5) [SUSv4]	lldiv(GLIBC_2.2.5) [SUSv4]
longjmp(GLIBC_2.2.5) [SUSv4]	lrand48(GLIBC_2.2.5) [SUSv4]	lrand48_r(GLIBC_2.2.5) [LSB]	lsearch(GLIBC_2.2.5) [SUSv4]
makecontext(GLIBC_2.2.5) [SUSv3]	malloc(GLIBC_2.2.5) [SUSv4]	memmem(GLIBC_2.2.5) [LSB]	mkdtemp(GLIBC_2.2.5) [SUSv4]
mkstemp(GLIBC_2.2.5) [SUSv4]	mktemp(GLIBC_2.2.5) [SUSv3]	mrnd48(GLIBC_2.2.5) [SUSv4]	mrnd48_r(GLIBC_2.2.5) [LSB]
nftw(GLIBC_2.3) [SUSv4]	nrnd48(GLIBC_2.2.5) [SUSv4]	nrnd48_r(GLIBC_2.2.5) [LSB]	ntohl(GLIBC_2.2.5) [SUSv4]
ntohs(GLIBC_2.2.5) [SUSv4]	open_memstream(GLIBC_2.2.5) [SUSv4]	openlog(GLIBC_2.2.5) [SUSv4]	perror(GLIBC_2.2.5) [SUSv4]
posix_openpt(GLIBC_2.2.5) [SUSv4]	ptsname(GLIBC_2.2.5) [SUSv4]	putenv(GLIBC_2.2.5) [SUSv4]	qsort(GLIBC_2.2.5) [SUSv4]
rand(GLIBC_2.2.5) [SUSv4]	rand_r(GLIBC_2.2.5) [SUSv4]	random(GLIBC_2.2.5) [SUSv4]	random_r(GLIBC_2.2.5) [LSB]
realloc(GLIBC_2.2.5) [SUSv4]	realpath(GLIBC_2.3) [SUSv4]	remque(GLIBC_2.2.5) [SUSv4]	scandir(GLIBC_2.2.5) [SUSv4]
scandir64(GLIBC_2.2.5) [LSB]	seed48(GLIBC_2.2.5) [SUSv4]	seed48_r(GLIBC_2.2.5) [LSB]	sendfile(GLIBC_2.2.5) [LSB]
setenv(GLIBC_2.2.5) [SUSv4]	sethostname(GLIBC_2.2.5) [LSB]	setlogmask(GLIBC_2.2.5) [SUSv4]	setstate(GLIBC_2.2.5) [SUSv4]
setstate_r(GLIBC_2.2.5) [LSB]	srand(GLIBC_2.2.5) [SUSv4]	srand48(GLIBC_2.2.5) [SUSv4]	srand48_r(GLIBC_2.2.5) [LSB]
srandom(GLIBC_2.2.5) [SUSv4]	srandom_r(GLIBC_2.2.5) [LSB]	strtod(GLIBC_2.2.5) [SUSv4]	strtol(GLIBC_2.2.5) [SUSv4]

strtoul(GLIBC_2.2.5) [SUSv4]	swapcontext(GLIBC_2.2.5) [SUSv3]	syslog(GLIBC_2.2.5) [SUSv4]	system(GLIBC_2.2.5) [LSB]
tdelete(GLIBC_2.2.5) [SUSv4]	tfind(GLIBC_2.2.5) [SUSv4]	tmpfile(GLIBC_2.2.5) [SUSv4]	tmpnam(GLIBC_2.2.5) [SUSv4]
tsearch(GLIBC_2.2.5) [SUSv4]	ttname(GLIBC_2.2.5) [SUSv4]	ttname_r(GLIBC_2.2.5) [SUSv4]	twalk(GLIBC_2.2.5) [SUSv4]
unlockpt(GLIBC_2.2.5) [SUSv4]	unsetenv(GLIBC_2.2.5) [SUSv4]	usleep(GLIBC_2.2.5) [SUSv3]	verrx(GLIBC_2.2.5) [LSB]
vfscanf(GLIBC_2.2.5) [LSB]	vscanf(GLIBC_2.2.5) [LSB]	vsscanf(GLIBC_2.2.5) [LSB]	vsyslog(GLIBC_2.2.5) [LSB]
warn(GLIBC_2.2.5) [LSB]	warnx(GLIBC_2.2.5) [LSB]	wordexp(GLIBC_2.2.5) [SUSv4]	wordfree(GLIBC_2.2.5) [SUSv4]

An LSB conforming implementation shall provide the architecture specific deprecated functions for Standard Library specified in [Table 10-33](#), with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 10-33 libc - Standard Library Deprecated Function Interfaces

basename(GLIBC_2.2.5) [LSB]	getdomainname(GLIBC_2.2.5) [LSB]	inet_aton(GLIBC_2.2.5) [LSB]	tmpnam(GLIBC_2.2.5) [SUSv4]
-----------------------------	----------------------------------	------------------------------	-----------------------------

An LSB conforming implementation shall provide the architecture specific data interfaces for Standard Library specified in [Table 10-34](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-34 libc - Standard Library Data Interfaces

__environ(GLIBC_2.2.5) [LSB]	_environ(GLIBC_2.2.5) [LSB]	_sys_errlist(GLIBC_2.12) [LSB]	environ(GLIBC_2.2.5) [SUSv4]
getdate_err(GLIBC_2.2.5) [SUSv4]	optarg(GLIBC_2.2.5) [SUSv4]	opterr(GLIBC_2.2.5) [SUSv4]	optind(GLIBC_2.2.5) [SUSv4]
optopt(GLIBC_2.2.5) [SUSv4]			

10.2.22 GNU Extensions for libc

10.2.22.1 Interfaces for GNU Extensions for libc

An LSB conforming implementation shall provide the architecture specific functions for GNU Extensions for libc specified in [Table 10-35](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-35 libc - GNU Extensions for libc Function Interfaces

gnu_get_libc_release(GLIBC_2.2.5) [LSB]	gnu_get_libc_version(GLIBC_2.2.5) [LSB]		
---	---	--	--

10.3 Data Definitions for libc

This section defines global identifiers and their values that are associated with interfaces contained in libc. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the

existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the [ISO C \(1999\)](#) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

10.3.1 argz.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.2 assert.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.3 cpio.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.4 ctype.h

```
enum {
    _ISupper = 256,
    _ISlower = 512,
    _ISalpha = 1024,
    _ISdigit = 2048,
    _ISxdigit = 4096,
    _ISspace = 8192,
    _ISprint = 16384,
    _ISgraph = 32768,
    _ISblank = 1,
    _IScntrl = 2,
    _ISpunct = 4,
    _ISalnum = 8
};
```

10.3.5 dirent.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.6 elf.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.7 endian.h

```

#define __BYTE_ORDER    __LITTLE_ENDIAN

```

10.3.8 errno.h

```

#define EDEADLOCK      EDEADLK

```

10.3.9 fcntl.h

```

#define O_LARGEFILE    0
#define O_DIRECTORY    0200000
#define O_NOFOLLOW     0400000
#define POSIX_FADV_DONTNEED  4
#define POSIX_FADV_NOREUSE   5

#define F_GETLK64      5
#define F_SETLK64      6
#define F_SETLKW64     7

```

10.3.10 fmtmsg.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.11 fnmatch.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.12 ftw.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.13 getopt.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.14 glob.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.15 iconv.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.16 ifaddrs.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.17 inttypes.h

```

#define __PRI64_PREFIX "l"
#define __PRIPTR_PREFIX "l"

typedef ldiv_t imaxdiv_t;

```

10.3.18 langinfo.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.19 limits.h

```

#define LONG_MAX          0x7FFFFFFFFFFFFFFFL
#define ULONG_MAX        0xFFFFFFFFFFFFFFFFUL
#define LONG_BIT          64

#define CHAR_MAX          127
#define CHAR_MIN          SCHAR_MIN

#define PTHREAD_STACK_MIN 16384

```

10.3.20 link.h

```

struct dl_phdr_info {
    Elf64_Addr dlpi_addr;
    const char *dlpi_name;
    const Elf64_Phdr *dlpi_phdr;
    Elf64_Half dlpi_phnum;
    unsigned long long int dlpi_adds;
    unsigned long long int dlpi_subs;
};

```



```

    size_t dlpi_tls_modid;
    void *dlpi_tls_data;
};

```

10.3.21 locale.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.22 lsb/time.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.23 lsb/types.h

```

typedef int64_t ssize_t;

```

10.3.24 lsb/wchar.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.25 net/if.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.26 net/db.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.27 netinet/icmp6.h

```

#define ND_NA_FLAG_OVERRIDE      0x00000020
#define ND_NA_FLAG_SOLICITED    0x00000040
#define ND_NA_FLAG_ROUTER       0x00000080
#define ICMP6_RR_RESULT_FLAGS_FORBIDDEN 0x0010
#define ICMP6_RR_RESULT_FLAGS_00B 0x0020

```

10.3.28 netinet/igmp.h

```

/*

```

```

* This header is architecture neutral
* Please refer to the generic specification for details
*/

```

10.3.29 netinet/in.h

```

/*
* This header is architecture neutral
* Please refer to the generic specification for details
*/

```

10.3.30 netinet/in_system.h

```

/*
* This header is architecture neutral
* Please refer to the generic specification for details
*/

```

10.3.31 netinet/ip.h

```

struct timestamp {
    u_int8_t len;
    u_int8_t ptr;
    unsigned int flags:4;
    unsigned int overflow:4;
    u_int32_t data[9];
};
struct iphdr {
    unsigned int ihl:4;
    unsigned int version:4;
    u_int8_t tos;
    u_int16_t tot_len;
    u_int16_t id;
    u_int16_t frag_off;
    u_int8_t ttl;
    u_int8_t protocol;
    u_int16_t check;
    u_int32_t saddr;
    u_int32_t daddr;
};
struct ip {
    unsigned int ip_hl:4;
    unsigned int ip_v:4;
    u_int8_t ip_tos;
    u_short ip_len;
    u_short ip_id;
    u_short ip_off;
    u_int8_t ip_ttl;
    u_int8_t ip_p;
    u_short ip_sum;
    struct in_addr ip_src;
    struct in_addr ip_dst;
};
struct ip_timestamp {
    u_int8_t ipt_code;
    u_int8_t ipt_len;
    u_int8_t ipt_ptr;
    unsigned int ipt_flg:4;
    unsigned int ipt_oflw:4;
    u_int32_t data[9];
};

```

10.3.32 netinet/ip6.h

```
#define IP6_ALERT_MLD    0x0000
#define IP6F_MORE_FRAG  0x0100
#define IP6_ALERT_RSVP  0x0100
#define IP6_ALERT_AN    0x0200
#define IP6F_RESERVED_MASK 0x0600
#define IP6F_OFF_MASK   0xf8ff
```

10.3.33 netinet/ip_icmp.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.34 netinet/tcp.h

```
struct tcphdr {
    uint16_t source;
    uint16_t dest;
    uint32_t seq;
    uint32_t ack_seq;
    uint16_t res1:4;
    uint16_t doff:4;
    uint16_t fin:1;
    uint16_t syn:1;
    uint16_t rst:1;
    uint16_t psh:1;
    uint16_t ack:1;
    uint16_t urg:1;
    uint16_t res2:2;
    uint16_t window;
    uint16_t check;
    uint16_t urg_ptr;
};
```

10.3.35 netinet/udp.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.36 nl_types.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.37 pwd.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.38 regex.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.39 rpc/auth.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.40 rpc/clnt.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.41 rpc/rpc_msg.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.42 rpc/svc.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.43 rpc/types.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.44 rpc/xdr.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.45 sched.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

```
*/
```

10.3.46 search.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.3.47 setjmp.h

```
typedef long int __jmp_buf[8];
```

10.3.48 signal.h

```
#define SIGEV_PAD_SIZE ((SIGEV_MAX_SIZE/sizeof(int))-4)

#define SI_PAD_SIZE ((SI_MAX_SIZE/sizeof(int))-4)

struct sigaction {
    union {
        sighandler_t _sa_handler;
        void (*_sa_sigaction) (int, siginfo_t *, void *);
    } __sigaction_handler;
    sigset_t sa_mask;
    int sa_flags;
    void (*sa_restorer) (void);
};

#define MINSIGSTKSZ 2048 /* Minimum stack size for a
signal handler. */
#define SIGSTKSZ 8192 /* System default stack size. */

struct _fpxreg {
    unsigned short significand[4];
    unsigned short exponent;
    unsigned short padding[3];
};

struct _xmmreg {
    uint32_t element[4];
};

struct _fpstate {
    uint16_t cwd;
    uint16_t swd;
    uint16_t ftw;
    uint16_t fop;
    uint64_t rip;
    uint64_t rdp;
    uint32_t mxcsr;
    uint32_t mxcr_mask;
    struct _fpxreg _st[8];
    struct _xmmreg _xmm[16];
    uint32_t padding[24];
};

struct sigcontext {
    unsigned long int r8;
    unsigned long int r9;
    unsigned long int r10;
    unsigned long int r11;
    unsigned long int r12;
```

```

unsigned long int r13;
unsigned long int r14;
unsigned long int r15;
unsigned long int rdi;
unsigned long int rsi;
unsigned long int rbp;
unsigned long int rbx;
unsigned long int rdx;
unsigned long int rax;
unsigned long int rcx;
unsigned long int rsp;
unsigned long int rip;
unsigned long int eflags;
unsigned short cs;
unsigned short gs;
unsigned short fs;
unsigned short __pad0;
unsigned long int err;
unsigned long int trapno;
unsigned long int oldmask;
unsigned long int cr2;
struct _fpstate *fpstate;
unsigned long int __reserved1[8];
};

```

10.3.49 spawn.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.50 stddef.h

```

typedef int wchar_t;
typedef unsigned long int size_t;
typedef long int ptrdiff_t;

```

10.3.51 stdint.h

```

#define INT64_C(c)          c ## L
#define INTMAX_C(c)        c ## L
#define __INT64_C(c)       c ## L
#define UINT64_C(c)        c ## UL
#define UINTMAX_C(c)       c ## UL
#define __UINT64_C(c)      c ## UL

#define INTPTR_MIN          (-9223372036854775807L-1)
#define INT_FAST16_MIN      (-9223372036854775807L-1)
#define INT_FAST32_MIN      (-9223372036854775807L-1)
#define PTRDIFF_MIN         (-9223372036854775807L-1)
#define SIZE_MAX            (18446744073709551615UL)
#define UINTPTR_MAX         (18446744073709551615UL)
#define INT_FAST16_MAX      (18446744073709551615UL)
#define INT_FAST32_MAX      (18446744073709551615UL)
#define INTPTR_MAX          (9223372036854775807L)
#define INT_FAST16_MAX      (9223372036854775807L)
#define INT_FAST32_MAX      (9223372036854775807L)
#define PTRDIFF_MAX         (9223372036854775807L)

typedef long int int64_t;
typedef long int intmax_t;

```

```

typedef unsigned long int uintmax_t;
typedef long int intptr_t;
typedef unsigned long int uintptr_t;
typedef unsigned long int uint64_t;
typedef long int int_least64_t;
typedef unsigned long int uint_least64_t;
typedef long int int_fast16_t;
typedef long int int_fast32_t;
typedef long int int_fast64_t;
typedef unsigned long int uint_fast16_t;
typedef unsigned long int uint_fast32_t;
typedef unsigned long int uint_fast64_t;

```

10.3.52 stdio.h

```
#define __IO_FILE_SIZE 216
```

10.3.53 stdlib.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.54 string.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.55 sys/epoll.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.56 sys/file.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.57 sys/inotify.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.58 sys/io.h

```
extern int ioperm(unsigned long int from, unsigned long int num,
```

```
        int turn_on);
extern int iopl(int level);
```

10.3.59 sys/ioctl.h

```
#define TIOCGWINSZ      0x5413
#define TIOC SWINSZ     0x5414
#define FIONREAD        0x541B
#define TIOCNOTTY      21538
```

10.3.60 sys/ipc.h

```
struct ipc_perm {
    key_t __key;
    uid_t uid;
    gid_t gid;
    uid_t cuid;
    uid_t cgid;
    unsigned short mode;
    unsigned short __pad1;
    unsigned short __seq;
    unsigned short __pad2;
    unsigned long int __unused1;
    unsigned long int __unused2;
};
```

10.3.61 sys/mman.h

```
#define MCL_CURRENT     1
#define MCL_FUTURE     2
```

10.3.62 sys/msg.h

```
typedef unsigned long int msgqnum_t;
typedef unsigned long int msglen_t;

struct msqid_ds {
    struct ipc_perm msg_perm; /* structure describing operation
    permission */
    time_t msg_stime; /* time of last msgsnd command */
    time_t msg_rtime; /* time of last msgrcv command */
    /*
    time_t msg_ctime; /* time of last change */
    unsigned long int __msg_cbytes; /* current number of
    bytes on queue */
    msgqnum_t msg_qnum; /* number of messages currently
    on queue */
    msglen_t msg_qbytes; /* max number of bytes allowed on
    queue */
    pid_t msg_lspid; /* pid of last msgsnd() */
    pid_t msg_lrpid; /* pid of last msgrcv() */
    unsigned long int __unused4;
    unsigned long int __unused5;
};
```

10.3.63 sys/param.h

```
/*
```



```

* This header is architecture neutral
* Please refer to the generic specification for details
*/

```

10.3.64 sys/poll.h

```

/*
* This header is architecture neutral
* Please refer to the generic specification for details
*/

```

10.3.65 sys/ptrace.h

```

enum __ptrace_request {
    PTRACE_TRACEME = 0,
    PTRACE_PEEKTEXT = 1,
    PTRACE_PEEKDATA = 2,
    PTRACE_PEEKUSER = 3,
    PTRACE_POKETEXT = 4,
    PTRACE_POKEDATA = 5,
    PTRACE_POKEUSER = 6,
    PTRACE_CONT = 7,
    PTRACE_KILL = 8,
    PTRACE_SINGLESTEP = 9,
    PTRACE_GETREGS = 12,
    PTRACE_SETREGS = 13,
    PTRACE_GETFPREGS = 14,
    PTRACE_SETFPREGS = 15,
    PTRACE_ATTACH = 16,
    PTRACE_DETACH = 17,
    PTRACE_GETFPXREGS = 18,
    PTRACE_SETFPXREGS = 19,
    PTRACE_SYSCALL = 24,
    PTRACE_SETOPTIONS = 0x4200,
    PTRACE_GETEVENTMSG = 0x4201,
    PTRACE_GETSIGINFO = 0x4202,
    PTRACE_SETSIGINFO = 0x4203
};

```

10.3.66 sys/resource.h

```

/*
* This header is architecture neutral
* Please refer to the generic specification for details
*/

```

10.3.67 sys/select.h

```

/*
* This header is architecture neutral
* Please refer to the generic specification for details
*/

```

10.3.68 sys/sem.h

```

struct semid_ds {
    struct ipc_perm sem_perm;
    time_t sem_otime;
};

```

```

    unsigned long int __unused1;
    time_t sem_ctime;
    unsigned long int __unused2;
    unsigned long int sem_nsems;
    unsigned long int __unused3;
    unsigned long int __unused4;
};

```

10.3.69 sys/shm.h

```

#define SHMLBA (__getpagesize())

typedef unsigned long int shmatt_t;

struct shmid_ds {
    struct ipc_perm shm_perm; /* operation permission struct */
    size_t shm_segsz; /* size of segment in bytes */
    time_t shm_atime; /* time of last shmat() */
    time_t shm_dtime; /* time of last shmdt() */
    time_t shm_ctime; /* time of last change by
shmctl() */
    pid_t shm_cpid; /* pid of creator */
    pid_t shm_lpid; /* pid of last shmop */
    shmatt_t shm_nattch; /* number of current attaches */
    unsigned long int __unused4;
    unsigned long int __unused5;
};

```

10.3.70 sys/socket.h

```

typedef uint64_t __ss_aligntype;

#define SO_RCVLOWAT 18
#define SO_SNDLOWAT 19
#define SO_RCVTIMEO 20
#define SO_SNDTIMEO 21

```

10.3.71 sys/stat.h

```

#define _MKNOD_VER 0
#define _STAT_VER 1

struct stat {
    dev_t st_dev;
    ino_t st_ino;
    nlink_t st_nlink;
    mode_t st_mode;
    uid_t st_uid;
    gid_t st_gid;
    int pad0;
    dev_t st_rdev;
    off_t st_size;
    blksize_t st_blksize;
    blkcnt_t st_blocks;
    struct timespec st_atim;
    struct timespec st_mtim;
    struct timespec st_ctim;
    unsigned long int __unused[3];
};

struct stat64 {
    dev_t st_dev;
    ino64_t st_ino;

```

```

nlink_t st_nlink;
mode_t st_mode;
uid_t st_uid;
gid_t st_gid;
int pad0;
dev_t st_rdev;
off_t st_size;
blksize_t st_blksize;
blkcnt64_t st_blocks;
struct timespec st_atim;
struct timespec st_mtim;
struct timespec st_ctim;
unsigned long int __unused[3];
};

```

10.3.72 sys/statfs.h

```

struct statfs {
    long int f_type;           /* type of filesystem */
    long int f_bsize;         /* optimal transfer block size */
    fsblkcnt_t f_blocks;      /* total data blocks in file
system */
    fsblkcnt_t f_bfree;       /* free blocks in fs */
    fsblkcnt_t f_bavail;      /* free blocks avail to non-
superuser */
    fsfilcnt_t f_files;       /* total file nodes in file
system */
    fsfilcnt_t f_ffree;       /* free file nodes in file system
*/
    fsid_t f_fsid;           /* file system id */
    long int f_namelen;       /* maximum length of filenames */
    long int f_frsize;        /* fragment size */
    long int f_spare[5];      /* spare for later */
};
struct statfs64 {
    long int f_type;           /* type of filesystem */
    long int f_bsize;         /* optimal transfer block size */
    fsblkcnt64_t f_blocks;    /* total data blocks in file
system */
    fsblkcnt64_t f_bfree;     /* free blocks in fs */
    fsblkcnt64_t f_bavail;    /* free blocks avail to non-
superuser */
    fsfilcnt64_t f_files;     /* total file nodes in file
system */
    fsfilcnt64_t f_ffree;     /* free file nodes in file system
*/
    fsid_t f_fsid;           /* file system id */
    long int f_namelen;       /* maximum length of filenames */
    long int f_frsize;        /* fragment size */
    long int f_spare[5];      /* spare for later */
};

```

10.3.73 sys/statvfs.h

```

struct statvfs {
    unsigned long int f_bsize;
    unsigned long int f_frsize;
    fsblkcnt_t f_blocks;
    fsblkcnt_t f_bfree;
    fsblkcnt_t f_bavail;
    fsfilcnt_t f_files;
    fsfilcnt_t f_ffree;
    fsfilcnt_t f_favail;
};

```

```

    unsigned long int f_fsid;
    unsigned long int f_flag;
    unsigned long int f_namemax;
    int __f_spare[6];
};
struct statvfs64 {
    unsigned long int f_bsize;
    unsigned long int f_frsize;
    fsblkcnt64_t f_blocks;
    fsblkcnt64_t f_bfree;
    fsblkcnt64_t f_bavail;
    fsfilcnt64_t f_files;
    fsfilcnt64_t f_ffree;
    fsfilcnt64_t f_favail;
    unsigned long int f_fsid;
    unsigned long int f_flag;
    unsigned long int f_namemax;
    int __f_spare[6];
};

```

10.3.74 sys/sysinfo.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.75 sys/time.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.76 sys/timeb.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.77 sys/times.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.78 sys/un.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.79 sys/utsname.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.80 sys/wait.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.81 sysexits.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.82 syslog.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.83 tar.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.84 termios.h

```

#define OLCUC      0000002
#define ONLCR     0000004
#define XCASE     0000004
#define NLDLY     0000400
#define CR1       0001000
#define IUCLC     0001000
#define CR2       0002000
#define CR3       0003000
#define CRDLY     0003000
#define TAB1      0004000
#define TAB2      0010000
#define TAB3      0014000
#define TABDLY    0014000
#define BS1       0020000
#define BSDLY     0020000
#define VT1       0040000
#define VTDLY     0040000
#define FF1       0100000
#define FFDLY     0100000

#define VSUSP     10
#define VEOL      11
#define VREPRINT  12

```

```

#define VDISCARD      13
#define VWERASE 14
#define VEOL2 16
#define VMIN 6
#define VSWTC 7
#define VSTART 8
#define VSTOP 9

#define IXON 0002000
#define IXOFF 0010000

#define CS6 0000020
#define CS7 0000040
#define CS8 0000060
#define CSIZE 0000060
#define CSTOPB 0000100
#define CREAD 0000200
#define PARENB 0000400
#define PARODD 0001000
#define HUPCL 0002000
#define CLOCAL 0004000
#define VTIME 5

#define ISIG 0000001
#define ICANON 0000002
#define ECHOE 0000020
#define ECHOK 0000040
#define ECHONL 0000100
#define NOFLSH 0000200
#define TOSTOP 0000400
#define ECHOCTL 0001000
#define ECHOPRT 0002000
#define ECHOKE 0004000
#define FLUSHO 0010000
#define PENDIN 0040000
#define IEXTEN 0100000

```

10.3.85 time.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.86 ucontext.h

```

struct _libc_fpxreg {
    unsigned short significand[4];
    unsigned short exponent;
    unsigned short padding[3];
};
struct _libc_xmmreg {
    uint32_t element[4];
};
enum {
    REG_R8 = 0,
    REG_R9 = 1,
    REG_R10 = 2,
    REG_R11 = 3,
    REG_R12 = 4,
    REG_R13 = 5,
    REG_R14 = 6,
    REG_R15 = 7,

```

```

REG_RDI = 8,
REG_RSI = 9,
REG_RBP = 10,
REG_RBX = 11,
REG_RDX = 12,
REG_RAX = 13,
REG_RCX = 14,
REG_RSP = 15,
REG_RIP = 16,
REG_EFL = 17,
REG_CSGSFS = 18,
REG_ERR = 19,
REG_TRAPNO = 20,
REG_OLDMASK = 21,
REG_CR2 = 22
};

typedef long int greg_t;

#define NGREG 23

typedef greg_t gregset_t[23];

struct _libc_fpstate {
    uint16_t cwd;
    uint16_t swd;
    uint16_t ftw;
    uint16_t fop;
    uint64_t rip;
    uint64_t rdp;
    uint32_t mxcsr;
    uint32_t mxcr_mask;
    struct _libc_fpxreg _st[8];
    struct _libc_xmmreg _xmm[16];
    uint32_t padding[24];
};
typedef struct _libc_fpstate *fpregset_t;

typedef struct {
    gregset_t gregs;
    fpregset_t fpregs;
    unsigned long int __reserved1[8];
} mcontext_t;

typedef struct ucontext {
    unsigned long int uc_flags;
    struct ucontext *uc_link;
    stack_t uc_stack;
    mcontext_t uc_mcontext;
    sigset_t uc_sigmask;
    struct _libc_fpstate __fpregs_mem;
} ucontext_t;

```

10.3.87 ulimit.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

10.3.88 unistd.h

```

/*

```

```

* This header is architecture neutral
* Please refer to the generic specification for details
*/

```

10.3.89 utime.h

```

/*
* This header is architecture neutral
* Please refer to the generic specification for details
*/

```

10.3.90 utmp.h

```

struct lastlog {
    int32_t ll_time;
    char ll_line[UT_LINESIZE];
    char ll_host[UT_HOSTSIZE];
};

struct utmp {
    short ut_type;           /* Type of login. */
    pid_t ut_pid;           /* Process ID of login process. */
    /*
    char ut_line[UT_LINESIZE]; /* Devicename. */
    char ut_id[4];           /* Inittab ID. */
    char ut_user[UT_NAMESIZE]; /* Username. */
    char ut_host[UT_HOSTSIZE]; /* Hostname for remote login. */
    struct exit_status ut_exit; /* Exit status of a process
marked as DEAD_PROCESS. */
    int ut_session;         /* Session ID, used for
windowing. */
    struct {
        int32_t tv_sec;
        int32_t tv_usec;
    } ut_tv;               /* Time entry was made. */
    int32_t ut_addr_v6[4]; /* Internet address of remote
host. */
    char __unused[20];     /* Reserved for future use. */
};

```

10.3.91 utmpx.h

```

struct utmpx {
    short ut_type;           /* Type of login. */
    pid_t ut_pid;           /* Process ID of login process. */
    /*
    char ut_line[UT_LINESIZE]; /* Devicename. */
    char ut_id[4];           /* Inittab ID. */
    char ut_user[UT_NAMESIZE]; /* Username. */
    char ut_host[UT_HOSTSIZE]; /* Hostname for remote login. */
    struct exit_status ut_exit; /* Exit status of a process
marked as DEAD_PROCESS. */
    int32_t ut_session;     /* Session ID, used for
windowing. */
    struct {
        int32_t tv_sec;     /* Seconds. */
        int32_t tv_usec;   /* Microseconds. */
    } ut_tv;               /* Time entry was made. */
    int32_t ut_addr_v6[4]; /* Internet address of remote
host. */
    char __unused[20];     /* Reserved for future use. */
};

```


10.3.92 wordexp.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.4 Interface Definitions for libc

The interfaces defined on the following pages are included in libc and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.

Other interfaces listed in [Section 10.2](#) shall behave as described in the referenced base document. For interfaces referencing LSB and not listed below, please see the generic part of the specification.

ioperm

Name

ioperm — set port input/output permissions

Synopsis

```
#include <sys/io.h> /* for glibc */

int ioperm(unsigned long from, unsigned long num, int turn_on);
```

Description

ioperm sets the port access permission bits for the process for num bytes starting from port address from to the value turn_on. The use of ioperm requires root privileges.

Only the first 0x3ff I/O ports can be specified in this manner. For more ports, the iopl function must be used. Permissions are not inherited on fork, but on exec they are. This is useful for giving port access permissions to non-privileged tasks.

Return Value

On success, zero is returned. On error, -1 is returned, and errno is set appropriately.

Notes

Libc5 treats it as a system call and has a prototype in <unistd.h>. Glibc1 does not have a prototype. Glibc2 has a prototype both in <sys/io.h> and in <sys/perm.h>. Avoid the latter, it is available on i386 only.

iopl

Name

iopl — change I/O privilege level

Synopsis

```
#include <sys/io.h> /* for glibc */
```

```
int iopl(int level);
```

Description

`ioctl` changes the I/O privilege level of the current process, as specified in `level`.

This call is necessary to allow 8514-compatible X servers to run under Linux. Since these X servers require access to all 65536 I/O ports, the `ioperm` call is not sufficient.

In addition to granting unrestricted I/O port access, running at a higher I/O privilege level also allows the process to disable interrupts. This will probably crash the system, and is not recommended.

Permissions are inherited by `fork` and `exec`.

The I/O privilege level for a normal process is 0.

Return Value

On success, zero is returned. On error, -1 is returned, and `errno` is set appropriately.

Errors

EINVAL

`level` is greater than 3.

EPERM

The current user is not the super-user.

10.5 Interfaces for libm

[Table 10-36](#) defines the library name and shared object name for the `libm` library

Table 10-36 libm Definition

Library:	libm
SONAME:	libm.so.6

The behavior of the interfaces in this library is specified by the following specifications:

[LSB] [LSB Core - Generic](#)

[SUSv3] [POSIX 1003.1-2001 \(ISO/IEC 9945-2003\)](#)

[SUSv4] [POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#)

10.5.1 Math

10.5.1.1 Interfaces for Math

An LSB conforming implementation shall provide the architecture specific functions for Math specified in [Table 10-37](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-37 libm - Math Function Interfaces

__finite(GLIBC_2.2.5) [LSB]	__finitef(GLIBC_2.2.5) [LSB]	__finitel(GLIBC_2.2.5) [LSB]	__fpclassify(GLIBC_2.2.5) [LSB]
__fpclassifyf(GLIBC_2.2.5) [LSB]	__fpclassifyl(GLIBC_2.2.5) [LSB]	__signbit(GLIBC_2.2.5) [LSB]	__signbitf(GLIBC_2.2.5) [LSB]
__signbitl(GLIBC_2.2.5) [LSB]	acos(GLIBC_2.2.5) [SUSv4]	acosf(GLIBC_2.2.5) [SUSv4]	acosh(GLIBC_2.2.5) [SUSv4]
acoshf(GLIBC_2.2.5) [SUSv4]	acoshl(GLIBC_2.2.5) [SUSv4]	acosl(GLIBC_2.2.5) [SUSv4]	asin(GLIBC_2.2.5) [SUSv4]

asin _f (GLIBC_2.2.5) [SUSv4]	asinh _f (GLIBC_2.2.5) [SUSv4]	asinh _f (GLIBC_2.2.5) [SUSv4]	asinh _l (GLIBC_2.2.5) [SUSv4]
asin _l (GLIBC_2.2.5) [SUSv4]	atan(GLIBC_2.2.5) [SUSv4]	atan2(GLIBC_2.2.5) [SUSv4]	atan2 _f (GLIBC_2.2.5) [SUSv4]
atan2 _l (GLIBC_2.2.5) [SUSv4]	atan _f (GLIBC_2.2.5) [SUSv4]	atanh(GLIBC_2.2.5) [SUSv4]	atanh _f (GLIBC_2.2.5) [SUSv4]
atanh _l (GLIBC_2.2.5) [SUSv4]	atan _l (GLIBC_2.2.5) [SUSv4]	cabs(GLIBC_2.2.5) [SUSv4]	cabs _f (GLIBC_2.2.5) [SUSv4]
cabs _l (GLIBC_2.2.5) [SUSv4]	cacos(GLIBC_2.2.5) [SUSv4]	cacos _f (GLIBC_2.2.5) [SUSv4]	cacosh(GLIBC_2.2.5) [SUSv4]
cacos _h _f (GLIBC_2.2.5) [SUSv4]	cacosh _l (GLIBC_2.2.5) [SUSv4]	cacos _l (GLIBC_2.2.5) [SUSv4]	carg(GLIBC_2.2.5) [SUSv4]
carg _f (GLIBC_2.2.5) [SUSv4]	carg _l (GLIBC_2.2.5) [SUSv4]	casin(GLIBC_2.2.5) [SUSv4]	casin _f (GLIBC_2.2.5) [SUSv4]
casinh(GLIBC_2.2.5) [SUSv4]	casinh _f (GLIBC_2.2.5) [SUSv4]	casinh _l (GLIBC_2.2.5) [SUSv4]	casin _l (GLIBC_2.2.5) [SUSv4]
catan(GLIBC_2.2.5) [SUSv4]	catan _f (GLIBC_2.2.5) [SUSv4]	catanh(GLIBC_2.2.5) [SUSv4]	catanh _f (GLIBC_2.2.5) [SUSv4]
catanh _l (GLIBC_2.2.5) [SUSv4]	catan _l (GLIBC_2.2.5) [SUSv4]	cbrt(GLIBC_2.2.5) [SUSv4]	cbrt _f (GLIBC_2.2.5) [SUSv4]
cbrt _l (GLIBC_2.2.5) [SUSv4]	ccos(GLIBC_2.2.5) [SUSv4]	ccos _f (GLIBC_2.2.5) [SUSv4]	ccosh(GLIBC_2.2.5) [SUSv4]
ccosh _f (GLIBC_2.2.5) [SUSv4]	ccosh _l (GLIBC_2.2.5) [SUSv4]	ccos _l (GLIBC_2.2.5) [SUSv4]	ceil(GLIBC_2.2.5) [SUSv4]
ceil _f (GLIBC_2.2.5) [SUSv4]	ceil _l (GLIBC_2.2.5) [SUSv4]	cexp(GLIBC_2.2.5) [SUSv4]	cexp _f (GLIBC_2.2.5) [SUSv4]
cexpl(GLIBC_2.2.5) [SUSv4]	cimag(GLIBC_2.2.5) [SUSv4]	cimag _f (GLIBC_2.2.5) [SUSv4]	cimag _l (GLIBC_2.2.5) [SUSv4]
clog(GLIBC_2.2.5) [SUSv4]	clog ₁₀ (GLIBC_2.2.5) [LSB]	clog ₁₀ _f (GLIBC_2.2.5) [LSB]	clog ₁₀ _l (GLIBC_2.2.5) [LSB]
clog _f (GLIBC_2.2.5) [SUSv4]	clog _l (GLIBC_2.2.5) [SUSv4]	conj(GLIBC_2.2.5) [SUSv4]	conj _f (GLIBC_2.2.5) [SUSv4]
conj _l (GLIBC_2.2.5) [SUSv4]	copysign(GLIBC_2.2.5) [SUSv4]	copysign _f (GLIBC_2.2.5) [SUSv4]	copysign _l (GLIBC_2.2.5) [SUSv4]
cos(GLIBC_2.2.5) [SUSv4]	cos _f (GLIBC_2.2.5) [SUSv4]	cosh(GLIBC_2.2.5) [SUSv4]	cosh _f (GLIBC_2.2.5) [SUSv4]
cosh _l (GLIBC_2.2.5) [SUSv4]	cos _l (GLIBC_2.2.5) [SUSv4]	cpow(GLIBC_2.2.5) [SUSv4]	cpow _f (GLIBC_2.2.5) [SUSv4]
cpow _l (GLIBC_2.2.5) [SUSv4]	cproj(GLIBC_2.2.5) [SUSv4]	cproj _f (GLIBC_2.2.5) [SUSv4]	cproj _l (GLIBC_2.2.5) [SUSv4]
creal(GLIBC_2.2.5) [SUSv4]	creal _f (GLIBC_2.2.5) [SUSv4]	creal _l (GLIBC_2.2.5) [SUSv4]	csin(GLIBC_2.2.5) [SUSv4]
csin _f (GLIBC_2.2.5) [SUSv4]	csinh(GLIBC_2.2.5) [SUSv4]	csinh _f (GLIBC_2.2.5) [SUSv4]	csinh _l (GLIBC_2.2.5) [SUSv4]
csin _l (GLIBC_2.2.5) [SUSv4]	csqrt(GLIBC_2.2.5) [SUSv4]	csqrt _f (GLIBC_2.2.5) [SUSv4]	csqrt _l (GLIBC_2.2.5) [SUSv4]
ctan(GLIBC_2.2.5) [SUSv4]	ctan _f (GLIBC_2.2.5) [SUSv4]	ctanh(GLIBC_2.2.5) [SUSv4]	ctanh _f (GLIBC_2.2.5) [SUSv4]
ctanh _l (GLIBC_2.2.5) [SUSv4]	ctan _l (GLIBC_2.2.5) [SUSv4]	drem(GLIBC_2.2.5) [LSB]	drem _f (GLIBC_2.2.5) [LSB]

LSB Core - X86-64 5.0

dreml(GLIBC_2.2.5) [LSB]	erf(GLIBC_2.2.5) [SUSv4]	erfc(GLIBC_2.2.5) [SUSv4]	erfcf(GLIBC_2.2.5) [SUSv4]
erfcl(GLIBC_2.2.5) [SUSv4]	erff(GLIBC_2.2.5) [SUSv4]	erfl(GLIBC_2.2.5) [SUSv4]	exp(GLIBC_2.2.5) [SUSv4]
exp10(GLIBC_2.2.5) [LSB]	exp10f(GLIBC_2.2.5) [LSB]	exp10l(GLIBC_2.2.5) [LSB]	exp2(GLIBC_2.2.5) [SUSv4]
exp2f(GLIBC_2.2.5) [SUSv4]	exp2l(GLIBC_2.2.5) [SUSv4]	expf(GLIBC_2.2.5) [SUSv4]	expl(GLIBC_2.2.5) [SUSv4]
expm1(GLIBC_2.2.5) [SUSv4]	expm1f(GLIBC_2.2.5) [SUSv4]	expm1l(GLIBC_2.2.5) [SUSv4]	fabs(GLIBC_2.2.5) [SUSv4]
fabsf(GLIBC_2.2.5) [SUSv4]	fabsl(GLIBC_2.2.5) [SUSv4]	fdim(GLIBC_2.2.5) [SUSv4]	fdimf(GLIBC_2.2.5) [SUSv4]
fdiml(GLIBC_2.2.5) [SUSv4]	feclearexcept(GLIBC_2.2.5) [SUSv4]	fedisablexcept(GLIBC_2.2.5) [LSB]	feenableexcept(GLIBC_2.2.5) [LSB]
fegetenv(GLIBC_2.2.5) [SUSv4]	fegetexcept(GLIBC_2.2.5) [LSB]	fegetexceptflag(GLIBC_2.2.5) [SUSv4]	fegetround(GLIBC_2.2.5) [SUSv4]
feholdexcept(GLIBC_2.2.5) [SUSv4]	feraiseexcept(GLIBC_2.2.5) [SUSv4]	fesetenv(GLIBC_2.2.5) [SUSv4]	fesetexceptflag(GLIBC_2.2.5) [SUSv4]
fesetround(GLIBC_2.2.5) [SUSv4]	fetestexcept(GLIBC_2.2.5) [SUSv4]	feupdateenv(GLIBC_2.2.5) [SUSv4]	finite(GLIBC_2.2.5) [LSB]
finitef(GLIBC_2.2.5) [LSB]	finitel(GLIBC_2.2.5) [LSB]	floor(GLIBC_2.2.5) [SUSv4]	floorf(GLIBC_2.2.5) [SUSv4]
floorl(GLIBC_2.2.5) [SUSv4]	fma(GLIBC_2.2.5) [SUSv4]	fmaf(GLIBC_2.2.5) [SUSv4]	fmal(GLIBC_2.2.5) [SUSv4]
fmax(GLIBC_2.2.5) [SUSv4]	fmaxf(GLIBC_2.2.5) [SUSv4]	fmaxl(GLIBC_2.2.5) [SUSv4]	fmin(GLIBC_2.2.5) [SUSv4]
fminf(GLIBC_2.2.5) [SUSv4]	fminl(GLIBC_2.2.5) [SUSv4]	fmod(GLIBC_2.2.5) [SUSv4]	fmodf(GLIBC_2.2.5) [SUSv4]
fmodl(GLIBC_2.2.5) [SUSv4]	frexp(GLIBC_2.2.5) [SUSv4]	frexpf(GLIBC_2.2.5) [SUSv4]	frexpl(GLIBC_2.2.5) [SUSv4]
gamma(GLIBC_2.2.5) [LSB]	gammaf(GLIBC_2.2.5) [LSB]	gammal(GLIBC_2.2.5) [LSB]	hypot(GLIBC_2.2.5) [SUSv4]
hypotf(GLIBC_2.2.5) [SUSv4]	hypotl(GLIBC_2.2.5) [SUSv4]	ilogb(GLIBC_2.2.5) [SUSv4]	ilogbf(GLIBC_2.2.5) [SUSv4]
ilogbl(GLIBC_2.2.5) [SUSv4]	j0(GLIBC_2.2.5) [SUSv4]	j0f(GLIBC_2.2.5) [LSB]	j0l(GLIBC_2.2.5) [LSB]
j1(GLIBC_2.2.5) [SUSv4]	j1f(GLIBC_2.2.5) [LSB]	j1l(GLIBC_2.2.5) [LSB]	jnl(GLIBC_2.2.5) [SUSv4]
jnf(GLIBC_2.2.5) [LSB]	jnl(GLIBC_2.2.5) [LSB]	ldexp(GLIBC_2.2.5) [SUSv4]	ldexpf(GLIBC_2.2.5) [SUSv4]
ldexpl(GLIBC_2.2.5) [SUSv4]	lgamma(GLIBC_2.2.5) [SUSv4]	lgamma_r(GLIBC_2.2.5) [LSB]	lgammaf(GLIBC_2.2.5) [SUSv4]
lgammaf_r(GLIBC_2.2.5) [LSB]	lgammal(GLIBC_2.2.5) [SUSv4]	lgammal_r(GLIBC_2.2.5) [LSB]	llrint(GLIBC_2.2.5) [SUSv4]
llrintf(GLIBC_2.2.5) [SUSv4]	llrintl(GLIBC_2.2.5) [SUSv4]	llround(GLIBC_2.2.5) [SUSv4]	llroundf(GLIBC_2.2.5) [SUSv4]
llroundl(GLIBC_2.2.5) [SUSv4]	log(GLIBC_2.2.5)	log10(GLIBC_2.2.5)	log10f(GLIBC_2.2.5)

.2.5) [SUSv4]	[SUSv4]	.5) [SUSv4]	2.5) [SUSv4]
log10l(GLIBC_2.2.5) [SUSv4]	log1p(GLIBC_2.2.5) [SUSv4]	log1pf(GLIBC_2.2.5) [SUSv4]	log1pl(GLIBC_2.2.5) [SUSv4]
log2(GLIBC_2.2.5) [SUSv4]	log2f(GLIBC_2.2.5) [SUSv4]	log2l(GLIBC_2.2.5) [SUSv4]	logb(GLIBC_2.2.5) [SUSv4]
logbf(GLIBC_2.2.5) [SUSv4]	logbl(GLIBC_2.2.5) [SUSv4]	logf(GLIBC_2.2.5) [SUSv4]	logl(GLIBC_2.2.5) [SUSv4]
lrint(GLIBC_2.2.5) [SUSv4]	lrintf(GLIBC_2.2.5) [SUSv4]	lrintl(GLIBC_2.2.5) [SUSv4]	lround(GLIBC_2.2.5) [SUSv4]
lroundf(GLIBC_2.2.5) [SUSv4]	lroundl(GLIBC_2.2.5) [SUSv4]	matherr(GLIBC_2.2.5) [LSB]	modf(GLIBC_2.2.5) [SUSv4]
modff(GLIBC_2.2.5) [SUSv4]	modfl(GLIBC_2.2.5) [SUSv4]	nan(GLIBC_2.2.5) [SUSv4]	nanf(GLIBC_2.2.5) [SUSv4]
nanl(GLIBC_2.2.5) [SUSv4]	nearbyint(GLIBC_2.2.5) [SUSv4]	nearbyintf(GLIBC_2.2.5) [SUSv4]	nearbyintl(GLIBC_2.2.5) [SUSv4]
nextafter(GLIBC_2.2.5) [SUSv4]	nextafterf(GLIBC_2.2.5) [SUSv4]	nextafterl(GLIBC_2.2.5) [SUSv4]	nexttoward(GLIBC_2.2.5) [SUSv4]
nexttowardf(GLIBC_2.2.5) [SUSv4]	nexttowardl(GLIBC_2.2.5) [SUSv4]	pow(GLIBC_2.2.5) [SUSv4]	pow10(GLIBC_2.2.5) [LSB]
pow10f(GLIBC_2.2.5) [LSB]	pow10l(GLIBC_2.2.5) [LSB]	powf(GLIBC_2.2.5) [SUSv4]	powl(GLIBC_2.2.5) [SUSv4]
remainder(GLIBC_2.2.5) [SUSv4]	remainderf(GLIBC_2.2.5) [SUSv4]	remainderl(GLIBC_2.2.5) [SUSv4]	remquo(GLIBC_2.2.5) [SUSv4]
remquof(GLIBC_2.2.5) [SUSv4]	remquol(GLIBC_2.2.5) [SUSv4]	rint(GLIBC_2.2.5) [SUSv4]	rintf(GLIBC_2.2.5) [SUSv4]
rintl(GLIBC_2.2.5) [SUSv4]	round(GLIBC_2.2.5) [SUSv4]	roundf(GLIBC_2.2.5) [SUSv4]	roundl(GLIBC_2.2.5) [SUSv4]
scalb(GLIBC_2.2.5) [SUSv3]	scalbf(GLIBC_2.2.5) [LSB]	scalbl(GLIBC_2.2.5) [LSB]	scalbln(GLIBC_2.2.5) [SUSv4]
scalblnf(GLIBC_2.2.5) [SUSv4]	scalblnl(GLIBC_2.2.5) [SUSv4]	scalbn(GLIBC_2.2.5) [SUSv4]	scalbnf(GLIBC_2.2.5) [SUSv4]
scalbnl(GLIBC_2.2.5) [SUSv4]	significand(GLIBC_2.2.5) [LSB]	significandf(GLIBC_2.2.5) [LSB]	significandl(GLIBC_2.2.5) [LSB]
sin(GLIBC_2.2.5) [SUSv4]	sincos(GLIBC_2.2.5) [LSB]	sincosf(GLIBC_2.2.5) [LSB]	sincosl(GLIBC_2.2.5) [LSB]
sinf(GLIBC_2.2.5) [SUSv4]	sinh(GLIBC_2.2.5) [SUSv4]	sinhf(GLIBC_2.2.5) [SUSv4]	sinhl(GLIBC_2.2.5) [SUSv4]
sinl(GLIBC_2.2.5) [SUSv4]	sqrt(GLIBC_2.2.5) [SUSv4]	sqrtf(GLIBC_2.2.5) [SUSv4]	sqrtl(GLIBC_2.2.5) [SUSv4]
tan(GLIBC_2.2.5) [SUSv4]	tanf(GLIBC_2.2.5) [SUSv4]	tanh(GLIBC_2.2.5) [SUSv4]	tanhf(GLIBC_2.2.5) [SUSv4]
tanhf(GLIBC_2.2.5) [SUSv4]	tanl(GLIBC_2.2.5) [SUSv4]	tgamma(GLIBC_2.2.5) [SUSv4]	tgammalf(GLIBC_2.2.5) [SUSv4]
tgammal(GLIBC_2.2.5) [SUSv4]	trunc(GLIBC_2.2.5) [SUSv4]	truncf(GLIBC_2.2.5) [SUSv4]	truncl(GLIBC_2.2.5) [SUSv4]
y0(GLIBC_2.2.5) [SUSv4]	y0f(GLIBC_2.2.5) [LSB]	y0l(GLIBC_2.2.5) [LSB]	y1(GLIBC_2.2.5) [SUSv4]
y1f(GLIBC_2.2.5) [LSB]	y1l(GLIBC_2.2.5) [LSB]	yn(GLIBC_2.2.5) [SUSv4]	ynf(GLIBC_2.2.5) [LSB]

ynl(GLIBC_2.2.5) [LSB]			
------------------------	--	--	--

An LSB conforming implementation shall provide the architecture specific deprecated functions for Math specified in [Table 10-38](#), with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 10-38 libm - Math Deprecated Function Interfaces

drem(GLIBC_2.2.5) [LSB]	dremf(GLIBC_2.2.5) [LSB]	dreml(GLIBC_2.2.5) [LSB]	finite(GLIBC_2.2.5) [LSB]
finitef(GLIBC_2.2.5) [LSB]	finitel(GLIBC_2.2.5) [LSB]	gamma(GLIBC_2.2.5) [LSB]	gammaf(GLIBC_2.2.5) [LSB]
gammal(GLIBC_2.2.5) [LSB]	matherr(GLIBC_2.2.5) [LSB]		

An LSB conforming implementation shall provide the architecture specific data interfaces for Math specified in [Table 10-39](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-39 libm - Math Data Interfaces

signgam(GLIBC_2.2.5) [SUSv4]			
------------------------------	--	--	--

10.6 Data Definitions for libm

This section defines global identifiers and their values that are associated with interfaces contained in libm. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the [ISO C \(1999\)](#) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

10.6.1 complex.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.6.2 fenv.h

```
#define FE_INVALID      0x01
#define FE_DIVBYZERO   0x04
#define FE_OVERFLOW    0x08
#define FE_UNDERFLOW   0x10
```

```

#define FE_INEXACT      0x20

#define FE_ALL_EXCEPT \
    (FE_INEXACT | FE_DIVBYZERO | FE_UNDERFLOW | FE_OVERFLOW | FE_INVALID)

#define FE_TONEAREST    0
#define FE_DOWNWARD     0x400
#define FE_UPWARD       0x800
#define FE_TOWARDZERO   0xc00

typedef unsigned short fexcept_t;

typedef struct {
    unsigned short __control_word;
    unsigned short __unused1;
    unsigned short __status_word;
    unsigned short __unused2;
    unsigned short __tags;
    unsigned short __unused3;
    unsigned int __eip;
    unsigned short __cs_selector;
    unsigned int __opcode:11;
    unsigned int __unused4:5;
    unsigned int __data_offset;
    unsigned short __data_selector;
    unsigned short __unused5;
    unsigned int __mxcsr;
} fenv_t;

#define FE_DFL_ENV      ((const fenv_t *) -1)

```

10.6.3 math.h

```

typedef float float_t;
typedef double double_t;

#define fpclassify(x) \
    (sizeof (x) == sizeof (float) ? __fpclassifyf (x) : sizeof (x) == sizeof (double) ? __fpclassify (x) : __fpclassifyl (x))
/* Return number of classification appropriate for X. */
#define signbit(x) \
    (sizeof (x) == sizeof (float)? __signbitf (x) : sizeof (x) == sizeof (double)? __signbit (x) : __signbitl (x)) /* Return nonzero value if sign of X is negative. */
#define isfinite(x) \
    (sizeof (x) == sizeof (float) ? __finitef (x) : sizeof (x) == sizeof (double)? __finite (x) : __finitel (x)) /* Return nonzero value if X is not +-Inf or NaN. */
#define isinf(x) \
    (sizeof (x) == sizeof (float) ? __isinf (x) : sizeof (x) == sizeof (double) ? __isinf (x) : __isinfl (x))
#define isnan(x) \
    (sizeof (x) == sizeof (float) ? __isnanf (x) : sizeof (x) == sizeof (double) ? __isnan (x) : __isnanl (x))

#define HUGE_VALL      0x1.0p32767L

#define FP_ILOGB0      -2147483648
#define FP_ILOGBNAN    -2147483648

extern int __fpclassifyl(long double);
extern int __signbitl(long double);
extern long double exp2l(long double);

```

10.7 Interface Definitions for libm

The interfaces defined on the following pages are included in libm and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.

Other interfaces listed in [Section 10.5](#) shall behave as described in the referenced base document. For interfaces referencing LSB and not listed below, please see the generic part of the specification.

__fpclassify

Name

`__fpclassify` — Classify real floating type

Synopsis

```
int __fpclassify(long double arg);
```

Description

`__fpclassify()` has the same specification as `fpclassify()` in [POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#), except that the argument type for `__fpclassify()` is known to be long double.

`__fpclassify()` is not in the source standard; it is only in the binary standard.

__signbitl

Name

`__signbitl` — test sign of floating point value

Synopsis

```
#include <math.h>
int __signbitl(long double arg);
```

Description

`__signbitl()` has the same specification as `signbit()` in [POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#), except that the argument type for `__signbitl()` is known to be long double.

`__signbitl()` is not in the source standard; it is only in the binary standard.

10.8 Interfaces for libpthread

[Table 10-40](#) defines the library name and shared object name for the libpthread library

Table 10-40 libpthread Definition

Library:	libpthread
SONAME:	libpthread.so.0

The behavior of the interfaces in this library is specified by the following specifications:

[LFS] [Large File Support](#)

[LSB] [LSB Core - Generic](#)

[SUSv3] [POSIX 1003.1-2001 \(ISO/IEC 9945-2003\)](#)

[SUSv4] [POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#)

10.8.1 Realtime Threads

10.8.1.1 Interfaces for Realtime Threads

An LSB conforming implementation shall provide the architecture specific functions for Realtime Threads specified in [Table 10-41](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-41 libpthread - Realtime Threads Function Interfaces

pthread_attr_getinheritsched(GLIBC_2.2.5) [SUSv4]	pthread_attr_getschedpolicy(GLIBC_2.2.5) [SUSv4]	pthread_attr_getscope(GLIBC_2.2.5) [SUSv4]	pthread_attr_setinheritsched(GLIBC_2.2.5) [SUSv4]
pthread_attr_setschedpolicy(GLIBC_2.2.5) [SUSv4]	pthread_attr_setscope(GLIBC_2.2.5) [SUSv4]	pthread_getschedparam(GLIBC_2.2.5) [SUSv4]	pthread_setschedparam(GLIBC_2.2.5) [SUSv4]

10.8.2 Advanced Realtime Threads

10.8.2.1 Interfaces for Advanced Realtime Threads

An LSB conforming implementation shall provide the architecture specific functions for Advanced Realtime Threads specified in [Table 10-42](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-42 libpthread - Advanced Realtime Threads Function Interfaces

pthread_barrier_destroy(GLIBC_2.2.5) [SUSv4]	pthread_barrier_init(GLIBC_2.2.5) [SUSv4]	pthread_barrier_wait(GLIBC_2.2.5) [SUSv4]	pthread_barrierattr_destroy(GLIBC_2.2.5) [SUSv4]
pthread_barrierattr_init(GLIBC_2.2.5) [SUSv4]	pthread_barrierattr_setshared(GLIBC_2.2.5) [SUSv4]	pthread_getcpuclockid(GLIBC_2.2.5) [SUSv4]	pthread_spin_destroy(GLIBC_2.2.5) [SUSv4]
pthread_spin_init(GLIBC_2.2.5) [SUSv4]	pthread_spin_lock(GLIBC_2.2.5) [SUSv4]	pthread_spin_trylock(GLIBC_2.2.5) [SUSv4]	pthread_spin_unlock(GLIBC_2.2.5) [SUSv4]

10.8.3 Posix Threads

10.8.3.1 Interfaces for Posix Threads

An LSB conforming implementation shall provide the architecture specific functions for Posix Threads specified in [Table 10-43](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-43 libpthread - Posix Threads Function Interfaces

_pthread_cleanup_pop(GLIBC_2.2.5) [LSB]	_pthread_cleanup_push(GLIBC_2.2.5) [LSB]	pthread_attr_destroy(GLIBC_2.2.5) [SUSv4]	pthread_attr_getdetachstate(GLIBC_2.2.5) [SUSv4]
pthread_attr_getguardsize(GLIBC_2.2.5) [SUSv4]	pthread_attr_getschedparam(GLIBC_2.2.5) [SUSv4]	pthread_attr_getstack(GLIBC_2.2.5) [SUSv4]	pthread_attr_getstackaddr(GLIBC_2.2.5) [SUSv3]
pthread_attr_getstacksize(GLIBC_2.2.5) [SUSv4]	pthread_attr_init(GLIBC_2.2.5) [SUSv4]	pthread_attr_setdetachstate(GLIBC_2.2.5) [SUSv4]	pthread_attr_setguardsize(GLIBC_2.2.5) [SUSv4]
pthread_attr_setschedparam(GLIBC_2.2.5) [SUSv4]	pthread_attr_setstack(GLIBC_2.2.5) [SUSv4]	pthread_attr_setstackaddr(GLIBC_2.2.5) [SUSv3]	pthread_attr_setstacksize(GLIBC_2.2.5) [SUSv4]

LSB Core - X86-64 5.0

pthread_cancel(GLIBC_2.2.5) [SUSv4]	pthread_cond_broadcast(GLIBC_2.3.2) [SUSv4]	pthread_cond_destroy(GLIBC_2.3.2) [SUSv4]	pthread_cond_init(GLIBC_2.3.2) [SUSv4]
pthread_cond_signal(GLIBC_2.3.2) [SUSv4]	pthread_cond_timedwait(GLIBC_2.3.2) [SUSv4]	pthread_cond_wait(GLIBC_2.3.2) [SUSv4]	pthread_condattr_destroy(GLIBC_2.2.5) [SUSv4]
pthread_condattr_getpshared(GLIBC_2.2.5) [SUSv4]	pthread_condattr_init(GLIBC_2.2.5) [SUSv4]	pthread_condattr_setpshared(GLIBC_2.2.5) [SUSv4]	pthread_create(GLIBC_2.2.5) [SUSv4]
pthread_detach(GLIBC_2.2.5) [SUSv4]	pthread_equal(GLIBC_2.2.5) [SUSv4]	pthread_exit(GLIBC_2.2.5) [SUSv4]	pthread_getconcurrency(GLIBC_2.2.5) [SUSv4]
pthread_getspecific(GLIBC_2.2.5) [SUSv4]	pthread_join(GLIBC_2.2.5) [SUSv4]	pthread_key_create(GLIBC_2.2.5) [SUSv4]	pthread_key_delete(GLIBC_2.2.5) [SUSv4]
pthread_kill(GLIBC_2.2.5) [SUSv4]	pthread_mutex_destroy(GLIBC_2.2.5) [SUSv4]	pthread_mutex_init(GLIBC_2.2.5) [SUSv4]	pthread_mutex_lock(GLIBC_2.2.5) [SUSv4]
pthread_mutex_timedlock(GLIBC_2.2.5) [SUSv4]	pthread_mutex_trylock(GLIBC_2.2.5) [SUSv4]	pthread_mutex_unlock(GLIBC_2.2.5) [SUSv4]	pthread_mutexattr_destroy(GLIBC_2.2.5) [SUSv4]
pthread_mutexattr_getpshared(GLIBC_2.2.5) [SUSv4]	pthread_mutexattr_gettype(GLIBC_2.2.5) [SUSv4]	pthread_mutexattr_init(GLIBC_2.2.5) [SUSv4]	pthread_mutexattr_setpshared(GLIBC_2.2.5) [SUSv4]
pthread_mutexattr_settype(GLIBC_2.2.5) [SUSv4]	pthread_once(GLIBC_2.2.5) [SUSv4]	pthread_rwlock_destroy(GLIBC_2.2.5) [SUSv4]	pthread_rwlock_init(GLIBC_2.2.5) [SUSv4]
pthread_rwlock_rdlock(GLIBC_2.2.5) [SUSv4]	pthread_rwlock_timedrdlock(GLIBC_2.2.5) [SUSv4]	pthread_rwlock_timedwrlock(GLIBC_2.2.5) [SUSv4]	pthread_rwlock_tryrdlock(GLIBC_2.2.5) [SUSv4]
pthread_rwlock_trywrlock(GLIBC_2.2.5) [SUSv4]	pthread_rwlock_unlock(GLIBC_2.2.5) [SUSv4]	pthread_rwlock_wrllock(GLIBC_2.2.5) [SUSv4]	pthread_rwlockattr_destroy(GLIBC_2.2.5) [SUSv4]
pthread_rwlockattr_getpshared(GLIBC_2.2.5) [SUSv4]	pthread_rwlockattr_init(GLIBC_2.2.5) [SUSv4]	pthread_rwlockattr_setpshared(GLIBC_2.2.5) [SUSv4]	pthread_self(GLIBC_2.2.5) [SUSv4]
pthread_setcancelstate(GLIBC_2.2.5) [SUSv4]	pthread_setcanceltype(GLIBC_2.2.5) [SUSv4]	pthread_setconcurrency(GLIBC_2.2.5) [SUSv4]	pthread_setspecific(GLIBC_2.2.5) [SUSv4]
pthread_sigmask(GLIBC_2.2.5) [SUSv4]	pthread_testcancel(GLIBC_2.2.5) [SUSv4]	sem_close(GLIBC_2.2.5) [SUSv4]	sem_destroy(GLIBC_2.2.5) [SUSv4]
sem_getvalue(GLIBC_2.2.5) [SUSv4]	sem_init(GLIBC_2.2.5) [SUSv4]	sem_open(GLIBC_2.2.5) [SUSv4]	sem_post(GLIBC_2.2.5) [SUSv4]
sem_timedwait(GLIBC_2.2.5) [SUSv4]	sem_trywait(GLIBC_2.2.5) [SUSv4]	sem_unlink(GLIBC_2.2.5) [SUSv4]	sem_wait(GLIBC_2.2.5) [SUSv4]

An LSB conforming implementation shall provide the architecture specific deprecated functions for Posix Threads specified in [Table 10-44](#), with the full mandatory functionality as described in the referenced underlying specification.

Note: These interfaces are deprecated, and applications should avoid using them. These interfaces may be withdrawn in future releases of this specification.

Table 10-44 libpthread - Posix Threads Deprecated Function Interfaces

pthread_attr_getst ackaddr(GLIBC_2 .2.5) [SUSv3]	pthread_attr_setsta ckaddr(GLIBC_2. 2.5) [SUSv3]		
--	--	--	--

10.8.4 Thread aware versions of libc interfaces

10.8.4.1 Interfaces for Thread aware versions of libc interfaces

An LSB conforming implementation shall provide the architecture specific functions for Thread aware versions of libc interfaces specified in [Table 10-45](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-45 libpthread - Thread aware versions of libc interfaces Function Interfaces

lseek64(GLIBC_2 .2.5) [LFS]	open64(GLIBC_2. 2.5) [LFS]	pread(GLIBC_2.2. 5) [SUSv4]	pread64(GLIBC_2 .2.5) [LSB]
pwrite(GLIBC_2. 2.5) [SUSv4]	pwrite64(GLIBC_ 2.2.5) [LSB]		

10.8.5 GNU Extensions for libpthread

10.8.5.1 Interfaces for GNU Extensions for libpthread

An LSB conforming implementation shall provide the architecture specific functions for GNU Extensions for libpthread specified in [Table 10-46](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-46 libpthread - GNU Extensions for libpthread Function Interfaces

pthread_getattr_np (GLIBC_2.2.5) [LSB]	pthread_mutex_co nsistent_np(GLIB C_2.4) [LSB]	pthread_mutexattr _getrobust_np(GL IBC_2.4) [LSB]	pthread_mutexattr _setrobust_np(GL IBC_2.4) [LSB]
pthread_rwlockatt r_getkind_np(GLI BC_2.2.5) [LSB]	pthread_rwlockatt r_setkind_np(GLI BC_2.2.5) [LSB]		

10.8.6 System Calls

10.8.6.1 Interfaces for System Calls

An LSB conforming implementation shall provide the architecture specific functions for System Calls specified in [Table 10-47](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-47 libpthread - System Calls Function Interfaces

close(GLIBC_2.2. 5) [SUSv4]	fcntl(GLIBC_2.2. 5) [LSB]	fork(GLIBC_2.2.5) [SUSv4]	fsync(GLIBC_2.2. 5) [SUSv4]
lseek(GLIBC_2.2. 5) [SUSv4]	msync(GLIBC_2. 2.5) [SUSv4]	nanosleep(GLIBC _2.2.5) [SUSv4]	open(GLIBC_2.2. 5) [SUSv4]
pause(GLIBC_2.2 .5) [SUSv4]	read(GLIBC_2.2.5) [SUSv4]	vfork(GLIBC_2.2. 5) [SUSv3]	wait(GLIBC_2.2.5) [SUSv4]
waitpid(GLIBC_2. 2.5) [LSB]	write(GLIBC_2.2. 5) [SUSv4]		

10.8.7 Standard I/O

10.8.7.1 Interfaces for Standard I/O

An LSB conforming implementation shall provide the architecture specific functions for Standard I/O specified in [Table 10-48](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-48 libpthread - Standard I/O Function Interfaces

flockfile(GLIBC_2.2.5) [SUSv4]			
--	--	--	--

10.8.8 Signal Handling

10.8.8.1 Interfaces for Signal Handling

An LSB conforming implementation shall provide the architecture specific functions for Signal Handling specified in [Table 10-49](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-49 libpthread - Signal Handling Function Interfaces

__libc_current_sigrtmax(GLIBC_2.2.5) [LSB]	__libc_current_sigrtmin(GLIBC_2.2.5) [LSB]	raise(GLIBC_2.2.5) [SUSv4]	sigaction(GLIBC_2.2.5) [SUSv4]
--	--	--	--

siglongjmp(GLIBC_2.2.5) [SUSv4]	sigwait(GLIBC_2.2.5) [SUSv4]		
---------------------------------	------------------------------	--	--

10.8.9 Standard Library

10.8.9.1 Interfaces for Standard Library

An LSB conforming implementation shall provide the architecture specific functions for Standard Library specified in [Table 10-50](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-50 libpthread - Standard Library Function Interfaces

__errno_location(GLIBC_2.2.5) [LSB]	ftwlockfile(GLIBC_2.2.5) [SUSv4]	funlockfile(GLIBC_2.2.5) [SUSv4]	longjmp(GLIBC_2.2.5) [SUSv4]
system(GLIBC_2.2.5) [LSB]			

10.8.10 Socket Interface

10.8.10.1 Interfaces for Socket Interface

An LSB conforming implementation shall provide the architecture specific functions for Socket Interface specified in [Table 10-51](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-51 libpthread - Socket Interface Function Interfaces

__h_errno_location(GLIBC_2.2.5) [LSB]	accept(GLIBC_2.2.5) [SUSv4]	connect(GLIBC_2.2.5) [SUSv4]	recv(GLIBC_2.2.5) [SUSv4]
recvfrom(GLIBC_2.2.5) [SUSv4]	recvmsg(GLIBC_2.2.5) [SUSv4]	send(GLIBC_2.2.5) [SUSv4]	sendmsg(GLIBC_2.2.5) [SUSv4]
sendto(GLIBC_2.2.5) [SUSv4]			

10.8.11 Terminal Interface Functions

10.8.11.1 Interfaces for Terminal Interface Functions

An LSB conforming implementation shall provide the architecture specific functions for Terminal Interface Functions specified in [Table 10-52](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-52 libpthread - Terminal Interface Functions Function Interfaces

tcdrain(GLIBC_2.2.5) [SUSv4]			
------------------------------	--	--	--

10.9 Data Definitions for libpthread

This section defines global identifiers and their values that are associated with interfaces contained in libpthread. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application

developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the [ISO C \(1999\)](#) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

10.9.1 lsb/pthread.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.9.2 pthread.h

```
#define __SIZEOF_PTHREAD_BARRIER_T      32
#define __SIZEOF_PTHREAD_MUTEX_T        40
#define __SIZEOF_PTHREAD_ATTR_T         56
#define __SIZEOF_PTHREAD_RWLOCK_T      56
#define PTHREAD_RWLOCK_INITIALIZER      { { 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0 } }
#define PTHREAD_MUTEX_INITIALIZER       { { 0, 0, 0, 0, 0, 0,
{ 0, 0 } } }

typedef union {
    char __size[__SIZEOF_PTHREAD_BARRIER_T];
    long int __align;
} pthread_barrier_t;

typedef struct __pthread_internal_list __pthread_list_t;
struct __pthread_mutex_s {
    int __lock;
    unsigned int __count;
    int __owner;
    unsigned int __nusers;
    int __kind;
    int __spins;
    __pthread_list_t __list;
};

typedef union {
    struct {
        int __lock;
        unsigned int __nr_readers;
        unsigned int __readers_wakeup;
        unsigned int __writer_wakeup;
        unsigned int __nr_readers_queued;
        unsigned int __nr_writers_queued;
        int __writer;
        int __pad1;
        unsigned long int __pad2;
        unsigned long int __pad3;
        unsigned int __flags;
    } __data;
    char __size[__SIZEOF_PTHREAD_RWLOCK_T];
    long int __align;
} pthread_rwlock_t;
```

10.9.3 semaphore.h

```
#define __SIZEOF_SEM_T 32
```

10.10 Interfaces for libgcc_s

[Table 10-53](#) defines the library name and shared object name for the libgcc_s library

Table 10-53 libgcc_s Definition

Library:	libgcc_s
SONAME:	libgcc_s.so.1

The behavior of the interfaces in this library is specified by the following specifications:
[LSB] [LSB Core - Generic](#)

10.10.1 Unwind Library

10.10.1.1 Interfaces for Unwind Library

An LSB conforming implementation shall provide the architecture specific functions for Unwind Library specified in [Table 10-54](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-54 libgcc_s - Unwind Library Function Interfaces

<code>_Unwind_Backtrace(GCC_3.3)</code> [LSB]	<code>_Unwind_DeleteException(GCC_3.0)</code> [LSB]	<code>_Unwind_FindEnclosingFunction(GCC_3.3)</code> [LSB]	<code>_Unwind_Find_FDE(GCC_3.0)</code> [LSB]
<code>_Unwind_ForcedUnwind(GCC_3.0)</code> [LSB]	<code>_Unwind_GetCFA(GCC_3.3)</code> [LSB]	<code>_Unwind_GetDataRelBase(GCC_3.0)</code> [LSB]	<code>_Unwind_GetGR(GCC_3.0)</code> [LSB]
<code>_Unwind_GetIP(GCC_3.0)</code> [LSB]	<code>_Unwind_GetLanguageSpecificData(GCC_3.0)</code> [LSB]	<code>_Unwind_GetRegionStart(GCC_3.0)</code> [LSB]	<code>_Unwind_GetTextRelBase(GCC_3.0)</code> [LSB]
<code>_Unwind_RaiseException(GCC_3.0)</code> [LSB]	<code>_Unwind_Resume(GCC_3.0)</code> [LSB]	<code>_Unwind_Resume_or_Rethrow(GCC_3.3)</code> [LSB]	<code>_Unwind_SetGR(GCC_3.0)</code> [LSB]
<code>_Unwind_SetIP(GCC_3.0)</code> [LSB]			

10.11 Data Definitions for libgcc_s

This section defines global identifiers and their values that are associated with interfaces contained in libgcc_s. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the [ISO C \(1999\)](#) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

10.11.1 unwind.h

```
extern _Unwind_Ptr _Unwind_GetDataRelBase(struct _Unwind_Context
*);
extern _Unwind_Ptr _Unwind_GetTextRelBase(struct _Unwind_Context
*);
```

10.12 Interface Definitions for libgcc_s

The interfaces defined on the following pages are included in libgcc_s and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.

Other interfaces listed in [Section 10.10](#) shall behave as described in the referenced base document. For interfaces referencing LSB and not listed below, please see the generic part of the specification.

_Unwind_Find_FDE

Name

`_Unwind_Find_FDE` — private C++ error handling method

Synopsis

```
fde * _Unwind_Find_FDE(void * pc, struct dwarf_eh_bases * bases);
```

Description

`_Unwind_Find_FDE()` looks for the object containing *pc*, then inserts into *bases*.

_Unwind_GetDataRelBase

Name

`_Unwind_GetDataRelBase` — private IA64 C++ error handling method

Synopsis

```
_Unwind_Ptr _Unwind_GetDataRelBase(struct _Unwind_Context * context);
```

Description

`_Unwind_GetDataRelBase()` returns the global pointer in register one for *context*.

_Unwind_GetTextRelBase

Name

`_Unwind_GetTextRelBase` — private IA64 C++ error handling method

Synopsis

```
_Unwind_Ptr _Unwind_GetTextRelBase(struct _Unwind_Context * context);
```

Description

`_Unwind_GetTextRelBase()` calls the abort method, then returns.

10.13 Interfaces for libdl

[Table 10-55](#) defines the library name and shared object name for the libdl library

Table 10-55 libdl Definition

Library:	libdl
SONAME:	libdl.so.2

The behavior of the interfaces in this library is specified by the following specifications:

[LSB] [LSB Core - Generic](#)

[SUSv4] [POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#)

10.13.1 Dynamic Loader

10.13.1.1 Interfaces for Dynamic Loader

An LSB conforming implementation shall provide the architecture specific functions for Dynamic Loader specified in [Table 10-56](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-56 libdl - Dynamic Loader Function Interfaces

dladdr(GLIBC_2.2.5) [LSB]	dlclose(GLIBC_2.2.5) [SUSv4]	dLError(GLIBC_2.2.5) [SUSv4]	dlopen(GLIBC_2.2.5) [LSB]
dlsym(GLIBC_2.2.5) [LSB]	dlvsym(GLIBC_2.2.5) [LSB]		

10.14 Data Definitions for libdl

This section defines global identifiers and their values that are associated with interfaces contained in libdl. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the [ISO C \(1999\)](#) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

10.14.1 dlfcn.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

10.15 Interfaces for libcrypt

[Table 10-57](#) defines the library name and shared object name for the libcrypt library

Table 10-57 libcrypt Definition

Library:	libcrypt
----------	----------

SONAME:	libcrypt.so.1
---------	---------------

The behavior of the interfaces in this library is specified by the following specifications:
 [LSB] [LSB Core - Generic](#)
 [SUSv4] [POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#)

10.15.1 Encryption

10.15.1.1 Interfaces for Encryption

An LSB conforming implementation shall provide the architecture specific functions for Encryption specified in [Table 10-58](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 10-58 libcrypt - Encryption Function Interfaces

crypt(GLIBC_2.2.5) [SUSv4]	crypt_r(GLIBC_2.2.5) [LSB]	encrypt(GLIBC_2.2.5) [SUSv4]	encrypt_r(GLIBC_2.2.5) [LSB]
setkey(GLIBC_2.2.5) [SUSv4]	setkey_r(GLIBC_2.2.5) [LSB]		

10.16 Data Definitions for libcrypt

This section defines global identifiers and their values that are associated with interfaces contained in libcrypt. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the [ISO C \(1999\)](#) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

10.16.1 crypt.h

```

/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */

```

IV Utility Libraries

11 Libraries

An LSB-conforming implementation shall also support the following utility libraries which are built on top of the interfaces provided by the base libraries. These libraries implement common functionality, and hide additional system dependent information such as file formats and device names.

11.1 Interfaces for libz

[Table 11-1](#) defines the library name and shared object name for the libz library

Table 11-1 libz Definition

Library:	libz
SONAME:	libz.so.1

11.1.1 Compression Library

11.1.1.1 Interfaces for Compression Library

No external functions are defined for libz - Compression Library in this part of the specification. See also the generic specification.

11.2 Data Definitions for libz

This section defines global identifiers and their values that are associated with interfaces contained in libz. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the [ISO C \(1999\)](#) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

11.2.1 zconf.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

11.2.2 zlib.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

11.3 Interfaces for libncurses

[Table 11-2](#) defines the library name and shared object name for the libncurses library

Table 11-2 libncurses Definition

Library:	libncurses
SONAME:	libncurses.so.5

11.3.1 Curses

11.3.1.1 Interfaces for Curses

No external functions are defined for libncurses - Curses in this part of the specification. See also the generic specification.

11.4 Data Definitions for libncurses

This section defines global identifiers and their values that are associated with interfaces contained in libncurses. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the [ISO C \(1999\)](#) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

11.4.1 curses.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

11.5 Interfaces for libncursesw

[Table 11-3](#) defines the library name and shared object name for the libncursesw library

Table 11-3 libncursesw Definition

Library:	libncursesw
SONAME:	libncursesw.so.5

11.5.1 Curses Wide

11.5.1.1 Interfaces for Curses Wide

No external functions are defined for libncursesw - Curses Wide in this part of the specification. See also the generic specification.

11.6 Data Definitions for libncursesw

This section defines global identifiers and their values that are associated with interfaces contained in libncursesw. These definitions are organized into groups that correspond to system headers. This convention is used as a convenience for the reader, and does not imply the existence of these headers, or their content. Where an interface is defined as requiring a particular system header file all of the data definitions for that system header

file presented here shall be in effect.

This section gives data definitions to promote binary application portability, not to repeat source interface definitions available elsewhere. System providers and application developers should use this ABI to supplement - not to replace - source interface definition specifications.

This specification uses the [ISO C \(1999\)](#) C Language as the reference programming language, and data definitions are specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of these data objects does not preclude their use by other programming languages.

11.6.1 ncursesw/curses.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

11.6.2 ncursesw/ncurses_dll.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

11.6.3 ncursesw/term.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

11.6.4 ncursesw/unctrl.h

```
/*
 * This header is architecture neutral
 * Please refer to the generic specification for details
 */
```

11.7 Interfaces for libutil

[Table 11-4](#) defines the library name and shared object name for the libutil library

Table 11-4 libutil Definition

Library:	libutil
SONAME:	libutil.so.1

The behavior of the interfaces in this library is specified by the following specifications:
[LSB] [LSB Core - Generic](#)

11.7.1 Utility Functions

11.7.1.1 Interfaces for Utility Functions

An LSB conforming implementation shall provide the architecture specific functions for Utility Functions specified in [Table 11-5](#), with the full mandatory functionality as de-

scribed in the referenced underlying specification.

Table 11-5 libutil - Utility Functions Function Interfaces

forkpty(GLIBC_2.2.5) [LSB]	login(GLIBC_2.2.5) [LSB]	login_tty(GLIBC_2.2.5) [LSB]	logout(GLIBC_2.2.5) [LSB]
logwtmp(GLIBC_2.2.5) [LSB]	openpty(GLIBC_2.2.5) [LSB]		

V Base Libraries

12 Libraries

An LSB-conforming implementation shall support some base libraries which provide interfaces for accessing the operating system, processor and other hardware in the system.

Interfaces that are unique to the AMD64 platform are defined here. This section should be used in conjunction with the corresponding section in the Linux Standard Base Specification.

12.1 Interfaces for libstdcxx

[Table 12-1](#) defines the library name and shared object name for the libstdcxx library

Table 12-1 libstdcxx Definition

Library:	libstdcxx
SONAME:	libstdc++.so.6

The behavior of the interfaces in this library is specified by the following specifications:

[CXXABI-1.86] [Itanium™ C++ ABI](#)
 [ISOCXX] [ISO/IEC 14882: 2003 C++ Language](#)
 [LSB] [LSB Core - Generic](#)

12.1.1 C++ Runtime Support

12.1.1.1 Interfaces for C++ Runtime Support

An LSB conforming implementation shall provide the architecture specific methods for C++ Runtime Support specified in [Table 12-2](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-2 libstdcxx - C++ Runtime Support Function Interfaces

<code>operator new[](unsigned long)(GLIBCXX_3.4)</code> [ISOCXX]
<code>operator new[](unsigned long, nothrow_t const&)(GLIBCXX_3.4)</code> [ISOCXX]
<code>operator new(unsigned long)(GLIBCXX_3.4)</code> [ISOCXX]
<code>operator new(unsigned long, nothrow_t const&)(GLIBCXX_3.4)</code> [ISOCXX]

12.1.2 C++ type descriptors for built-in types

12.1.2.1 Interfaces for C++ type descriptors for built-in types

No external methods are defined for libstdcxx - C++ type descriptors for built-in types in this part of the specification. See also the generic specification.

12.1.3 C++ `_Rb_tree`

12.1.3.1 Interfaces for C++ `_Rb_tree`

No external methods are defined for libstdcxx - C++ `_Rb_tree` in this part of the specification. See also the generic specification.

12.1.4 Class `type_info`

12.1.4.1 Class data for `type_info`

The virtual table for the `std::type_info` class is described in the generic part of this specification.

The Run Time Type Information for the `std::type_info` class is described by [Table 12-3](#)

Table 12-3 typeinfo for `type_info`

Base Vtable	vtable for <code>__cxxabiv1::__class_type_info</code>
Name	typeinfo name for <code>type_info</code>

12.1.4.2 Interfaces for Class `type_info`

No external methods are defined for libstdc++ - Class `std::type_info` in this part of the specification. See also the generic specification.

12.1.5 Class `__cxxabiv1::__enum_type_info`**12.1.5.1 Class data for `__cxxabiv1::__enum_type_info`**

The virtual table for the `__cxxabiv1::__enum_type_info` class is described in the generic part of this specification.

The Run Time Type Information for the `__cxxabiv1::__enum_type_info` class is described by [Table 12-4](#)

Table 12-4 typeinfo for `__cxxabiv1::__enum_type_info`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>__cxxabiv1::__enum_type_info</code>

12.1.5.2 Interfaces for Class `__cxxabiv1::__enum_type_info`

No external methods are defined for libstdc++ - Class `__cxxabiv1::__enum_type_info` in this part of the specification. See also the generic specification.

12.1.6 Class `__cxxabiv1::__array_type_info`**12.1.6.1 Class data for `__cxxabiv1::__array_type_info`**

The virtual table for the `__cxxabiv1::__array_type_info` class is described in the generic part of this specification.

The Run Time Type Information for the `__cxxabiv1::__array_type_info` class is described by [Table 12-5](#)

Table 12-5 typeinfo for `__cxxabiv1::__array_type_info`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>__cxxabiv1::__array_type_info</code>

12.1.6.2 Interfaces for Class `__cxxabiv1::__array_type_info`

No external methods are defined for libstdc++ - Class `__cxxabiv1::__array_type_info` in this part of the specification. See also the generic specification.

12.1.7 Class `__cxxabiv1::__class_type_info`**12.1.7.1 Class data for `__cxxabiv1::__class_type_info`**

The virtual table for the `__cxxabiv1::__class_type_info` class is described by [Table 12-6](#)

Table 12-6 Primary vtable for `__cxxabiv1::__class_type_info`

Base Offset	0
-------------	---

Virtual Base Offset	0
RTTI	typeid for __cxxabiv1::__class_type_info
vfunc[0]:	__cxxabiv1::__class_type_info::~~__class_type_info()
vfunc[1]:	__cxxabiv1::__class_type_info::~~__class_type_info()
vfunc[2]:	type_info::__is_pointer_p() const
vfunc[3]:	type_info::__is_function_p() const
vfunc[4]:	__cxxabiv1::__class_type_info::__do_catch(type_info const*, void**, unsigned int) const
vfunc[5]:	__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const
vfunc[6]:	__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&) const
vfunc[7]:	__cxxabiv1::__class_type_info::__do_dynacast(long, __cxxabiv1::__class_type_info::__sub_kind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__dynacast_result&) const
vfunc[8]:	__cxxabiv1::__class_type_info::__do_find_public_src(long, void const*, __cxxabiv1::__class_type_info const*, void const*) const

The Run Time Type Information for the __cxxabiv1::__class_type_info class is described by [Table 12-7](#)

Table 12-7 typeid for __cxxabiv1::__class_type_info

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for __cxxabiv1::__class_type_info

12.1.7.2 Interfaces for Class __cxxabiv1::__class_type_info

An LSB conforming implementation shall provide the architecture specific methods for Class __cxxabiv1::__class_type_info specified in [Table 12-8](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-8 libstdc++ - Class __cxxabiv1::__class_type_info Function Interfaces

__cxxabiv1::__class_type_info::__do_dynacast(long, __cxxabiv1::__class_type_info::__sub_kind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__dynacast_result&) const(CXXABI_1.3)
--

[CXXABI-1.86]

<code>__cxxabiv1::__class_type_info::__do_find_public_src(long, void const*, __cxxabiv1::__class_type_info const*, void const*) const(CXXABI_1.3) [CXXABI-1.86]</code>
--

12.1.8 Class `__cxxabiv1::__pbase_type_info`

12.1.8.1 Class data for `__cxxabiv1::__pbase_type_info`

The virtual table for the `__cxxabiv1::__pbase_type_info` class is described in the generic part of this specification.

The Run Time Type Information for the `__cxxabiv1::__pbase_type_info` class is described by [Table 12-9](#)

Table 12-9 typeinfo for `__cxxabiv1::__pbase_type_info`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>__cxxabiv1::__pbase_type_info</code>

12.1.8.2 Interfaces for Class `__cxxabiv1::__pbase_type_info`

No external methods are defined for libstdc++ - Class `__cxxabiv1::__pbase_type_info` in this part of the specification. See also the generic specification.

12.1.9 Class `__cxxabiv1::__pointer_type_info`

12.1.9.1 Class data for `__cxxabiv1::__pointer_type_info`

The virtual table for the `__cxxabiv1::__pointer_type_info` class is described in the generic part of this specification.

The Run Time Type Information for the `__cxxabiv1::__pointer_type_info` class is described by [Table 12-10](#)

Table 12-10 typeinfo for `__cxxabiv1::__pointer_type_info`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>__cxxabiv1::__pointer_type_info</code>

12.1.9.2 Interfaces for Class `__cxxabiv1::__pointer_type_info`

No external methods are defined for libstdc++ - Class `__cxxabiv1::__pointer_type_info` in this part of the specification. See also the generic specification.

12.1.10 Class `__cxxabiv1::__function_type_info`

12.1.10.1 Class data for `__cxxabiv1::__function_type_info`

The virtual table for the `__cxxabiv1::__function_type_info` class is described in the generic part of this specification.

The Run Time Type Information for the `__cxxabiv1::__function_type_info` class is described by [Table 12-11](#)

Table 12-11 typeinfo for `__cxxabiv1::__function_type_info`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
-------------	---

Name	typeinfo name for __cxxabiv1::__function_type_info
------	---

12.1.10.2 Interfaces for Class __cxxabiv1::__function_type_info

No external methods are defined for libstdc++ - Class __cxxabiv1::__function_type_info in this part of the specification. See also the generic specification.

12.1.11 Class __cxxabiv1::__si_class_type_info

12.1.11.1 Class data for __cxxabiv1::__si_class_type_info

The virtual table for the __cxxabiv1::__si_class_type_info class is described by [Table 12-12](#)

Table 12-12 Primary vtable for __cxxabiv1::__si_class_type_info

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for __cxxabiv1::__si_class_type_info
vfunc[0]:	__cxxabiv1::__si_class_type_info::~~__si_class_type_info()
vfunc[1]:	__cxxabiv1::__si_class_type_info::~~__si_class_type_info()
vfunc[2]:	type_info::__is_pointer_p() const
vfunc[3]:	type_info::__is_function_p() const
vfunc[4]:	__cxxabiv1::__class_type_info::__do_catch(type_info const*, void**, unsigned int) const
vfunc[5]:	__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const
vfunc[6]:	__cxxabiv1::__si_class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&) const
vfunc[7]:	__cxxabiv1::__si_class_type_info::__do_dyncast(long, __cxxabiv1::__class_type_info::__sub_kind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__dyncast_result&) const
vfunc[8]:	__cxxabiv1::__si_class_type_info::__do_find_public_src(long, void const*, __cxxabiv1::__class_type_info const*, void const*) const

The Run Time Type Information for the __cxxabiv1::__si_class_type_info class is described by [Table 12-13](#)

Table 12-13 typeinfo for `__cxxabiv1::__si_class_type_info`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>__cxxabiv1::__si_class_type_info</code>

12.1.11.2 Interfaces for Class `__cxxabiv1::__si_class_type_info`

An LSB conforming implementation shall provide the architecture specific methods for Class `__cxxabiv1::__si_class_type_info` specified in [Table 12-14](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-14 `libstdc++` - Class `__cxxabiv1::__si_class_type_info` Function Interfaces

<code>__cxxabiv1::__si_class_type_info::__do_dyncast(long, __cxxabiv1::__class_type_info::__sub_kind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__dyncast_result&) const</code> (CXXABI_1.3) [CXXABI-1.86]
<code>__cxxabiv1::__si_class_type_info::__do_find_public_src(long, void const*, __cxxabiv1::__class_type_info const*, void const*) const</code> (CXXABI_1.3) [CXXABI-1.86]

12.1.12 Class `__cxxabiv1::__vmi_class_type_info`**12.1.12.1 Class data for `__cxxabiv1::__vmi_class_type_info`**

The virtual table for the `__cxxabiv1::__vmi_class_type_info` class is described by [Table 12-15](#)

Table 12-15 Primary vtable for `__cxxabiv1::__vmi_class_type_info`

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>__cxxabiv1::__vmi_class_type_info</code>
<code>vfunc[0]:</code>	<code>__cxxabiv1::__vmi_class_type_info::~~__vmi_class_type_info()</code>
<code>vfunc[1]:</code>	<code>__cxxabiv1::__vmi_class_type_info::~~__vmi_class_type_info()</code>
<code>vfunc[2]:</code>	<code>type_info::__is_pointer_p() const</code>
<code>vfunc[3]:</code>	<code>type_info::__is_function_p() const</code>
<code>vfunc[4]:</code>	<code>__cxxabiv1::__class_type_info::__do_catch(type_info const*, void**, unsigned int) const</code>
<code>vfunc[5]:</code>	<code>__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</code>
<code>vfunc[6]:</code>	<code>__cxxabiv1::__vmi_class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&) const</code>
<code>vfunc[7]:</code>	<code>__cxxabiv1::__vmi_class_type_info::__do_dyncast(long, __cxxabiv1::__class_type_info::__sub_k</code>

	ind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__dynca st_result&) const
vfunc[8]:	__cxxabiv1::__vmi_class_type_info::__do_find_public_src(long, void const*, __cxxabiv1::__class_type_info const*, void const*) const

The Run Time Type Information for the __cxxabiv1::__vmi_class_type_info class is described by [Table 12-16](#)

Table 12-16 typeinfo for __cxxabiv1::__vmi_class_type_info

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for __cxxabiv1::__vmi_class_type_info

12.1.12.2 Interfaces for Class __cxxabiv1::__vmi_class_type_info

An LSB conforming implementation shall provide the architecture specific methods for Class __cxxabiv1::__vmi_class_type_info specified in [Table 12-17](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-17 libstdcxx - Class __cxxabiv1::__vmi_class_type_info Function Interfaces

__cxxabiv1::__vmi_class_type_info::__do_dyncast(long, __cxxabiv1::__class_type_info::__sub_kind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__dyncast_result&) const(CXXABI_1.3) [CXXABI-1.86]
__cxxabiv1::__vmi_class_type_info::__do_find_public_src(long, void const*, __cxxabiv1::__class_type_info const*, void const*) const(CXXABI_1.3) [CXXABI-1.86]

12.1.13 Class __cxxabiv1::__fundamental_type_info

12.1.13.1 Class data for __cxxabiv1::__fundamental_type_info

The virtual table for the __cxxabiv1::__fundamental_type_info class is described in the generic part of this specification.

The Run Time Type Information for the __cxxabiv1::__fundamental_type_info class is described by [Table 12-18](#)

Table 12-18 typeinfo for __cxxabiv1::__fundamental_type_info

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for __cxxabiv1::__fundamental_type_info

12.1.13.2 Interfaces for Class __cxxabiv1::__fundamental_type_info

No external methods are defined for libstdcxx - Class __cxxabiv1::__fundamental_type_info in this part of the specification. See also the generic specification.

12.1.14 Class

`__cxxabiv1::__pointer_to_member_type_info`

12.1.14.1 Class data for

`__cxxabiv1::__pointer_to_member_type_info`

The virtual table for the `__cxxabiv1::__pointer_to_member_type_info` class is described in the generic part of this specification.

The Run Time Type Information for the `__cxxabiv1::__pointer_to_member_type_info` class is described by [Table 12-19](#)

Table 12-19 typeinfo for `__cxxabiv1::__pointer_to_member_type_info`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>__cxxabiv1::__pointer_to_member_type_info</code>

12.1.14.2 Interfaces for Class

`__cxxabiv1::__pointer_to_member_type_info`

No external methods are defined for `libstdc++` - Class `__cxxabiv1::__pointer_to_member_type_info` in this part of the specification. See also the generic specification.

12.1.15 Class `__gnu_cxx::stdio_filebuf<char, char_traits<char> >`

12.1.15.1 Interfaces for Class `__gnu_cxx::stdio_filebuf<char, char_traits<char> >`

No external methods are defined for `libstdc++` - Class `__gnu_cxx::stdio_filebuf<char, std::char_traits<char> >` in this part of the specification. See also the generic specification.

12.1.16 Class `__gnu_cxx::stdio_filebuf<wchar_t, char_traits<wchar_t> >`

12.1.16.1 Interfaces for Class `__gnu_cxx::stdio_filebuf<wchar_t, char_traits<wchar_t> >`

No external methods are defined for `libstdc++` - Class `__gnu_cxx::stdio_filebuf<wchar_t, std::char_traits<wchar_t> >` in this part of the specification. See also the generic specification.

12.1.17 Class `__gnu_cxx::__pool_alloc_base`

12.1.17.1 Interfaces for Class `__gnu_cxx::__pool_alloc_base`

An LSB conforming implementation shall provide the architecture specific methods for Class `__gnu_cxx::__pool_alloc_base` specified in [Table 12-20](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-20 `libstdc++` - Class `__gnu_cxx::__pool_alloc_base` Function Interfaces

<code>__gnu_cxx::__pool_alloc_base::M_get_free_list(unsigned long)(GLIBCXX_3.4.2)</code> [LSB]
<code>__gnu_cxx::__pool_alloc_base::M_refill(unsigned long)(GLIBCXX_3.4.2)</code> [LSB]

12.1.18 Class `__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>>`

12.1.18.1 Class data for `__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>>`

The virtual table for the `__gnu_cxx::stdio_sync_filebuf<char, std::char_traits<char>>` class is described by [Table 12-21](#)

Table 12-21 Primary vtable for `__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>>`

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code>
vfunc[0]:	<code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code> <code>>::~stdio_sync_filebuf()</code>
vfunc[1]:	<code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code> <code>>::~stdio_sync_filebuf()</code>
vfunc[2]:	<code>basic_streambuf<char, char_traits<char>></code> <code>>::imbue(locale const&)</code>
vfunc[3]:	<code>basic_streambuf<char, char_traits<char>></code> <code>>::setbuf(char*, long)</code>
vfunc[4]:	<code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code> <code>>::seekoff(long, _Ios_Seekdir, _Ios_Openmode)</code>
vfunc[5]:	<code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code> <code>>::seekpos(fpos<__mbstate_t>, _Ios_Openmode)</code>
vfunc[6]:	<code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code> <code>>::sync()</code>
vfunc[7]:	<code>basic_streambuf<char, char_traits<char>></code> <code>>::showmanyc()</code>
vfunc[8]:	<code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code> <code>>::xsgetn(char*, long)</code>
vfunc[9]:	<code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code> <code>>::underflow()</code>
vfunc[10]:	<code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code> <code>>::uflow()</code>
vfunc[11]:	<code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code> <code>>::pbackfail(int)</code>
vfunc[12]:	<code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code> <code>>::xsputn(char const*, long)</code>
vfunc[13]:	<code>__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>></code> <code>>::overflow(int)</code>

12.1.18.2 Interfaces for Class `__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>>`

No external methods are defined for `libstdcxx` - Class `__gnu_cxx::stdio_sync_filebuf<char, std::char_traits<char>>` in this part of the specification. See also the generic specification.

12.1.19 Class

`__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>`

12.1.19.1 Class data for `__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>`

The virtual table for the `__gnu_cxx::stdio_sync_filebuf<wchar_t, std::char_traits<wchar_t>>` class is described by [Table 12-22](#)

Table 12-22 Primary vtable for `__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>`

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>></code>
vfunc[0]:	<code>__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>::~~stdio_sync_filebuf()</code>
vfunc[1]:	<code>__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>::~~stdio_sync_filebuf()</code>
vfunc[2]:	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::imbue(locale const&)</code>
vfunc[3]:	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::setbuf(wchar_t*, long)</code>
vfunc[4]:	<code>__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>::seekoff(long, _Ios_Seekdir, _Ios_Openmode)</code>
vfunc[5]:	<code>__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>::seekpos(fpos<__mbstate_t>, _Ios_Openmode)</code>
vfunc[6]:	<code>__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>::sync()</code>
vfunc[7]:	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::showmanyc()</code>
vfunc[8]:	<code>__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>::xsgetn(wchar_t*, long)</code>
vfunc[9]:	<code>__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>::underflow()</code>
vfunc[10]:	<code>__gnu_cxx::stdio_sync_filebuf<wchar_t,</code>

	char_traits<wchar_t>>::uflow()
vfunc[11]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::pbackfail(unsigned int)
vfunc[12]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::xsputn(wchar_t const*, long)
vfunc[13]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::overflow(unsigned int)

12.1.19.2 Interfaces for Class

__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >

No external methods are defined for libstdc++ - Class __gnu_cxx::stdio_sync_filebuf<wchar_t, std::char_traits<wchar_t> > in this part of the specification. See also the generic specification.

12.1.20 Class exception

12.1.20.1 Class data for exception

The virtual table for the std::exception class is described in the generic part of this specification.

The Run Time Type Information for the std::exception class is described by [Table 12-23](#)

Table 12-23 typeinfo for exception

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for exception

12.1.20.2 Interfaces for Class exception

No external methods are defined for libstdc++ - Class std::exception in this part of the specification. See also the generic specification.

12.1.21 Class bad_typeid

12.1.21.1 Class data for bad_typeid

The virtual table for the std::bad_typeid class is described in the generic part of this specification.

The Run Time Type Information for the std::bad_typeid class is described by [Table 12-24](#)

Table 12-24 typeinfo for bad_typeid

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for bad_typeid

12.1.21.2 Interfaces for Class bad_typeid

No external methods are defined for libstdc++ - Class std::bad_typeid in this part of the specification. See also the generic specification.

12.1.22 Class `logic_error`

12.1.22.1 Class data for `logic_error`

The virtual table for the `std::logic_error` class is described in the generic part of this specification.

The Run Time Type Information for the `std::logic_error` class is described by [Table 12-25](#)

Table 12-25 typeinfo for `logic_error`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>logic_error</code>

12.1.22.2 Interfaces for Class `logic_error`

No external methods are defined for `libstdc++` - Class `std::logic_error` in this part of the specification. See also the generic specification.

12.1.23 Class `range_error`

12.1.23.1 Class data for `range_error`

The virtual table for the `std::range_error` class is described in the generic part of this specification.

The Run Time Type Information for the `std::range_error` class is described by [Table 12-26](#)

Table 12-26 typeinfo for `range_error`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>range_error</code>

12.1.23.2 Interfaces for Class `range_error`

No external methods are defined for `libstdc++` - Class `std::range_error` in this part of the specification. See also the generic specification.

12.1.24 Class `domain_error`

12.1.24.1 Class data for `domain_error`

The virtual table for the `std::domain_error` class is described in the generic part of this specification.

The Run Time Type Information for the `std::domain_error` class is described by [Table 12-27](#)

Table 12-27 typeinfo for `domain_error`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>domain_error</code>

12.1.24.2 Interfaces for Class `domain_error`

No external methods are defined for `libstdc++` - Class `std::domain_error` in this part of the specification. See also the generic specification.

12.1.25 Class length_error

12.1.25.1 Class data for length_error

The virtual table for the std::length_error class is described in the generic part of this specification.

The Run Time Type Information for the std::length_error class is described by [Table 12-28](#)

Table 12-28 typeid for length_error

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for length_error

12.1.25.2 Interfaces for Class length_error

No external methods are defined for libstdc++ - Class std::length_error in this part of the specification. See also the generic specification.

12.1.26 Class out_of_range

12.1.26.1 Class data for out_of_range

The virtual table for the std::out_of_range class is described in the generic part of this specification.

The Run Time Type Information for the std::out_of_range class is described by [Table 12-29](#)

Table 12-29 typeid for out_of_range

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for out_of_range

12.1.26.2 Interfaces for Class out_of_range

No external methods are defined for libstdc++ - Class std::out_of_range in this part of the specification. See also the generic specification.

12.1.27 Class bad_exception

12.1.27.1 Class data for bad_exception

The virtual table for the std::bad_exception class is described in the generic part of this specification.

The Run Time Type Information for the std::bad_exception class is described by [Table 12-30](#)

Table 12-30 typeid for bad_exception

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for bad_exception

12.1.27.2 Interfaces for Class bad_exception

No external methods are defined for libstdc++ - Class std::bad_exception in this part of the specification. See also the generic specification.

12.1.28 Class runtime_error

12.1.28.1 Class data for runtime_error

The virtual table for the `std::runtime_error` class is described in the generic part of this specification.

The Run Time Type Information for the `std::runtime_error` class is described by [Table 12-31](#)

Table 12-31 typeinfo for runtime_error

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>runtime_error</code>

12.1.28.2 Interfaces for Class runtime_error

No external methods are defined for `libstdc++` - Class `std::runtime_error` in this part of the specification. See also the generic specification.

12.1.29 Class overflow_error

12.1.29.1 Class data for overflow_error

The virtual table for the `std::overflow_error` class is described in the generic part of this specification.

The Run Time Type Information for the `std::overflow_error` class is described by [Table 12-32](#)

Table 12-32 typeinfo for overflow_error

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>overflow_error</code>

12.1.29.2 Interfaces for Class overflow_error

No external methods are defined for `libstdc++` - Class `std::overflow_error` in this part of the specification. See also the generic specification.

12.1.30 Class underflow_error

12.1.30.1 Class data for underflow_error

The virtual table for the `std::underflow_error` class is described in the generic part of this specification.

The Run Time Type Information for the `std::underflow_error` class is described by [Table 12-33](#)

Table 12-33 typeinfo for underflow_error

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>underflow_error</code>

12.1.30.2 Interfaces for Class underflow_error

No external methods are defined for `libstdc++` - Class `std::underflow_error` in this part of the specification. See also the generic specification.

12.1.31 Class `invalid_argument`

12.1.31.1 Class data for `invalid_argument`

The virtual table for the `std::invalid_argument` class is described in the generic part of this specification.

The Run Time Type Information for the `std::invalid_argument` class is described by [Table 12-34](#)

Table 12-34 `typeinfo` for `invalid_argument`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	<code>typeinfo</code> name for <code>invalid_argument</code>

12.1.31.2 Interfaces for Class `invalid_argument`

No external methods are defined for `libstdcxx` - Class `std::invalid_argument` in this part of the specification. See also the generic specification.

12.1.32 Class `bad_cast`

12.1.32.1 Class data for `bad_cast`

The virtual table for the `std::bad_cast` class is described in the generic part of this specification.

The Run Time Type Information for the `std::bad_cast` class is described by [Table 12-35](#)

Table 12-35 `typeinfo` for `bad_cast`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	<code>typeinfo</code> name for <code>bad_cast</code>

12.1.32.2 Interfaces for Class `bad_cast`

No external methods are defined for `libstdcxx` - Class `std::bad_cast` in this part of the specification. See also the generic specification.

12.1.33 Class `bad_alloc`

12.1.33.1 Class data for `bad_alloc`

The virtual table for the `std::bad_alloc` class is described in the generic part of this specification.

The Run Time Type Information for the `std::bad_alloc` class is described by [Table 12-36](#)

Table 12-36 `typeinfo` for `bad_alloc`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	<code>typeinfo</code> name for <code>bad_alloc</code>

12.1.33.2 Interfaces for Class `bad_alloc`

No external methods are defined for `libstdcxx` - Class `std::bad_alloc` in this part of the specification. See also the generic specification.

12.1.34 struct `__numeric_limits_base`

12.1.34.1 Interfaces for struct `__numeric_limits_base`

No external methods are defined for `libstdcxx` - struct `__numeric_limits_base` in this part of the specification. See also the generic specification.

12.1.35 struct `numeric_limits<long double>`

12.1.35.1 Interfaces for struct `numeric_limits<long double>`

No external methods are defined for `libstdcxx` - struct `numeric_limits<long double>` in this part of the specification. See also the generic specification.

12.1.36 struct `numeric_limits<long long>`

12.1.36.1 Interfaces for struct `numeric_limits<long long>`

No external methods are defined for `libstdcxx` - struct `numeric_limits<long long>` in this part of the specification. See also the generic specification.

12.1.37 struct `numeric_limits<unsigned long long>`

12.1.37.1 Interfaces for struct `numeric_limits<unsigned long long>`

No external methods are defined for `libstdcxx` - struct `numeric_limits<unsigned long long>` in this part of the specification. See also the generic specification.

12.1.38 struct `numeric_limits<float>`

12.1.38.1 Interfaces for struct `numeric_limits<float>`

No external methods are defined for `libstdcxx` - struct `numeric_limits<float>` in this part of the specification. See also the generic specification.

12.1.39 struct `numeric_limits<double>`

12.1.39.1 Interfaces for struct `numeric_limits<double>`

No external methods are defined for `libstdcxx` - struct `numeric_limits<double>` in this part of the specification. See also the generic specification.

12.1.40 struct `numeric_limits<short>`

12.1.40.1 Interfaces for struct `numeric_limits<short>`

No external methods are defined for `libstdcxx` - struct `numeric_limits<short>` in this part of the specification. See also the generic specification.

12.1.41 struct `numeric_limits<unsigned short>`

12.1.41.1 Interfaces for struct `numeric_limits<unsigned short>`

No external methods are defined for `libstdcxx` - struct `numeric_limits<unsigned short>` in this part of the specification. See also the generic specification.

12.1.42 struct `numeric_limits<int>`

12.1.42.1 Interfaces for struct `numeric_limits<int>`

No external methods are defined for `libstdcxx` - struct `numeric_limits<int>` in this part

of the specification. See also the generic specification.

12.1.43 struct numeric_limits<unsigned int>

12.1.43.1 Interfaces for struct numeric_limits<unsigned int>

No external methods are defined for libstdcxx - struct numeric_limits<unsigned int> in this part of the specification. See also the generic specification.

12.1.44 struct numeric_limits<long>

12.1.44.1 Interfaces for struct numeric_limits<long>

No external methods are defined for libstdcxx - struct numeric_limits<long> in this part of the specification. See also the generic specification.

12.1.45 struct numeric_limits<unsigned long>

12.1.45.1 Interfaces for struct numeric_limits<unsigned long>

No external methods are defined for libstdcxx - struct numeric_limits<unsigned long> in this part of the specification. See also the generic specification.

12.1.46 struct numeric_limits<wchar_t>

12.1.46.1 Interfaces for struct numeric_limits<wchar_t>

No external methods are defined for libstdcxx - struct numeric_limits<wchar_t> in this part of the specification. See also the generic specification.

12.1.47 struct numeric_limits<unsigned char>

12.1.47.1 Interfaces for struct numeric_limits<unsigned char>

No external methods are defined for libstdcxx - struct numeric_limits<unsigned char> in this part of the specification. See also the generic specification.

12.1.48 struct numeric_limits<signed char>

12.1.48.1 Interfaces for struct numeric_limits<signed char>

No external methods are defined for libstdcxx - struct numeric_limits<signed char> in this part of the specification. See also the generic specification.

12.1.49 struct numeric_limits<char>

12.1.49.1 Interfaces for struct numeric_limits<char>

No external methods are defined for libstdcxx - struct numeric_limits<char> in this part of the specification. See also the generic specification.

12.1.50 struct numeric_limits<bool>

12.1.50.1 Interfaces for struct numeric_limits<bool>

No external methods are defined for libstdcxx - struct numeric_limits<bool> in this part of the specification. See also the generic specification.

12.1.51 Class `ctype_base`

12.1.51.1 Class data for `ctype_base`

The Run Time Type Information for the `std::ctype_base` class is described by [Table 12-37](#)

Table 12-37 `typeinfo` for `ctype_base`

Base Vtable	vtable for <code>__cxxabiv1::__class_type_info</code>
Name	<code>typeinfo</code> name for <code>ctype_base</code>

12.1.51.2 Interfaces for Class `ctype_base`

No external methods are defined for `libstdcxx` - Class `std::ctype_base` in this part of the specification. See also the generic specification.

12.1.52 Class `__ctype_abstract_base<char>`

12.1.52.1 Class data for `__ctype_abstract_base<char>`

The virtual table for the `std::__ctype_abstract_base<char>` class is described in the generic part of this specification.

12.1.52.2 Interfaces for Class `__ctype_abstract_base<char>`

No external methods are defined for `libstdcxx` - Class `std::__ctype_abstract_base<char>` in this part of the specification. See also the generic specification.

12.1.53 Class `__ctype_abstract_base<wchar_t>`

12.1.53.1 Class data for `__ctype_abstract_base<wchar_t>`

The virtual table for the `std::__ctype_abstract_base<wchar_t>` class is described in the generic part of this specification.

12.1.53.2 Interfaces for Class `__ctype_abstract_base<wchar_t>`

No external methods are defined for `libstdcxx` - Class `std::__ctype_abstract_base<wchar_t>` in this part of the specification. See also the generic specification.

12.1.54 Class `ctype<char>`

12.1.54.1 Class data for `ctype<char>`

The virtual table for the `std::ctype<char>` class is described in the generic part of this specification.

12.1.54.2 Interfaces for Class `ctype<char>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::ctype<char>` specified in [Table 12-38](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-38 `libstdcxx` - Class `ctype<char>` Function Interfaces

<code>ctype<char>::ctype(__locale_struct*, unsigned short const*, bool, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype<char>::ctype(unsigned short const*, bool, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]

ctype<char>::ctype(__locale_struct*, unsigned short const*, bool, unsigned long) (GLIBCXX_3.4) [ISOCXX]
ctype<char>::ctype(unsigned short const*, bool, unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.55 Class ctype<wchar_t>

12.1.55.1 Class data for ctype<wchar_t>

The virtual table for the std::ctype<wchar_t> class is described in the generic part of this specification.

The Run Time Type Information for the std::ctype<wchar_t> class is described by [Table 12-39](#)

Table 12-39 typeinfo for ctype<wchar_t>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for ctype<wchar_t>

12.1.55.2 Interfaces for Class ctype<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::ctype<wchar_t> specified in [Table 12-40](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-40 libstdcxx - Class ctype<wchar_t> Function Interfaces

ctype<wchar_t>::ctype(__locale_struct*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::ctype(unsigned long)(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::ctype(__locale_struct*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::ctype(unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.56 Class ctype_byname<char>

12.1.56.1 Class data for ctype_byname<char>

The virtual table for the std::ctype_byname<char> class is described in the generic part of this specification.

The Run Time Type Information for the std::ctype_byname<char> class is described by [Table 12-41](#)

Table 12-41 typeinfo for ctype_byname<char>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for ctype_byname<char>

12.1.56.2 Interfaces for Class ctype_byname<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::ctype_byname<char> specified in [Table 12-42](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-42 libstdcxx - Class ctype_byname<char> Function Interfaces

ctype_byname<char>::ctype_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
ctype_byname<char>::ctype_byname(char const*, unsigned long)(GLIBCXX_3.4)

[\[ISOCXX\]](#)

12.1.57 Class `ctype_byname<wchar_t>`

12.1.57.1 Class data for `ctype_byname<wchar_t>`

The virtual table for the `std::ctype_byname<wchar_t>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::ctype_byname<wchar_t>` class is described by [Table 12-43](#)

Table 12-43 typeinfo for `ctype_byname<wchar_t>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>ctype_byname<wchar_t></code>

12.1.57.2 Interfaces for Class `ctype_byname<wchar_t>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::ctype_byname<wchar_t>` specified in [Table 12-44](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-44 `libstdcxx` - Class `ctype_byname<wchar_t>` Function Interfaces

<code>ctype_byname<wchar_t>::ctype_byname(char const*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype_byname<wchar_t>::ctype_byname(char const*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]

12.1.58 Class `basic_string<char, char_traits<char>, allocator<char>>`

12.1.58.1 Interfaces for Class `basic_string<char, char_traits<char>, allocator<char>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::basic_string<char, std::char_traits<char>, std::allocator<char>>` specified in [Table 12-45](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-45 `libstdcxx` - Class `basic_string<char, char_traits<char>, allocator<char>>` Function Interfaces

<code>basic_string<char, char_traits<char>, allocator<char>>::find_last_of(char const*, unsigned long) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string<char, char_traits<char>, allocator<char>>::find_last_of(char const*, unsigned long, unsigned long) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string<char, char_traits<char>, allocator<char>>::find_last_of(basic_string<char, char_traits<char>, allocator<char>> const&, unsigned long) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string<char, char_traits<char>, allocator<char>>::find_last_of(char, unsigned long) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string<char, char_traits<char>, allocator<char>>::find_first_of(char const*, unsigned long) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string<char, char_traits<char>, allocator<char>>::find_first_of(char const*, unsigned long, unsigned long) const</code> (GLIBCXX_3.4) [ISOCXX]

<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find_first_of(basic_string<char, char_traits<char>, allocator<char>> const&</code> <code>, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find_first_of(char, unsigned</code> <code>long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::M_check_length(unsigned</code> <code>long, unsigned long, char const*) const(GLIBCXX_3.4.5) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find_last_not_of(char</code> <code>const*, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find_last_not_of(char</code> <code>const*, unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find_last_not_of(basic_string<char, char_traits<char>, allocator<char>></code> <code> const&, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find_last_not_of(char,</code> <code>unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find_first_not_of(char</code> <code>const*, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find_first_not_of(char</code> <code>const*, unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find_first_not_of(basic_string<char, char_traits<char>, allocator<char>></code> <code> const&, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find_first_not_of(char,</code> <code>unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::at(unsigned long)</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::copy(char*, unsigned long,</code> <code>unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find(char const*, unsigned</code> <code>long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find(char const*, unsigned</code> <code>long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find(basic_string<char,</code> <code>char_traits<char>, allocator<char>> const&, unsigned long) const(GLIBCXX_3.4)</code> <code>[ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::find(char, unsigned long)</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::rfind(char const*, unsigned</code> <code>long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::rfind(char const*, unsigned</code> <code>long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::rfind(basic_string<char,</code> <code>char_traits<char>, allocator<char>> const&, unsigned long) const(GLIBCXX_3.4)</code> <code>[ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::rfind(char, unsigned long)</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::substr(unsigned long,</code> <code>unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code></p>
<p><code>basic_string<char, char_traits<char>, allocator<char>></code> <code>>::compare(unsigned long,</code></p>

unsigned long, char const*) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::compare(unsigned long, unsigned long, char const*, unsigned long) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::compare(unsigned long, unsigned long, basic_string<char, char_traits<char>, allocator<char> > const&) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::compare(unsigned long, unsigned long, basic_string<char, char_traits<char>, allocator<char> > const&, unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_check(unsigned long, char const*) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_limit(unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::operator[](unsigned long) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_S_construct(unsigned long, char, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_replace_aux(unsigned long, unsigned long, unsigned long, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_replace_safe(unsigned long, unsigned long, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::at(unsigned long) (GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_Rep::_M_set_length_and_sharable(unsigned long)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_Rep::_M_clone(allocator<char> const&, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_Rep::_S_create(unsigned long, unsigned long, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::erase(unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::append(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::append(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned long, unsigned long) (GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::append(unsigned long, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::assign(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::assign(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned long, unsigned long) (GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::assign(unsigned long, char) (GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_insert(__gnu_cxx::__normal_iterator<char*, basic_string<char, char_traits<char>, allocator<char> >, unsigned long, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::insert(unsigned long, char

const*)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::insert(unsigned long, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::insert(unsigned long, basic_string<char, char_traits<char>, allocator<char> > const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::insert(unsigned long, basic_string<char, char_traits<char>, allocator<char> > const&, unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::insert(unsigned long, unsigned long, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::resize(unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::resize(unsigned long, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_copy(char*, char const*, unsigned long)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_move(char*, char const*, unsigned long)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(__gnu_cxx::__normal_iterator<char*, basic_string<char, char_traits<char>, allocator<char> > >, __gnu_cxx::__normal_iterator<char*, basic_string<char, char_traits<char>, allocator<char> > >, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(__gnu_cxx::__normal_iterator<char*, basic_string<char, char_traits<char>, allocator<char> > >, __gnu_cxx::__normal_iterator<char*, basic_string<char, char_traits<char>, allocator<char> > >, unsigned long, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(unsigned long, unsigned long, char const*)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(unsigned long, unsigned long, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(unsigned long, unsigned long, basic_string<char, char_traits<char>, allocator<char> > const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(unsigned long, unsigned long, unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(unsigned long, unsigned long, unsigned long, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::reserve(unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_assign(char*, unsigned long, char)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_mutate(unsigned long, unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::basic_string(char const*, unsigned long, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::basic_string(basic_string<char, char_traits<char>, allocator<char> > const&,

<code>unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<char, char_traits<char>, allocator<char>> >::basic_string(basic_string<char, char_traits<char>, allocator<char>> const&, unsigned long, unsigned long, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<char, char_traits<char>, allocator<char>> >::basic_string(unsigned long, char, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<char, char_traits<char>, allocator<char>> >::basic_string(char const* unsigned long, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<char, char_traits<char>, allocator<char>> >::basic_string(basic_string<char, char_traits<char>, allocator<char>> const&, unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<char, char_traits<char>, allocator<char>> >::basic_string(basic_string<char, char_traits<char>, allocator<char>> const&, unsigned long, unsigned long, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<char, char_traits<char>, allocator<char>> >::basic_string(unsigned long, char, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<char, char_traits<char>, allocator<char>> >::operator[](unsigned long) (GLIBCXX_3.4) [ISOCXX]</code>

12.1.59 Class `basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

12.1.59.1 Interfaces for Class `basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::basic_string<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>>` specified in [Table 12-46](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-46 libstdcxx - Class `basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>` Function Interfaces

<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >::find_last_of(wchar_t const*, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >::find_last_of(wchar_t const*, unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >::find_last_of(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >::find_last_of(wchar_t, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >::find_first_of(wchar_t const*, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >::find_first_of(wchar_t const*, unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >::find_first_of(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >::find_first_of(wchar_t, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code>

<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::M_check_length(unsigned long, unsigned long, char const*) const(GLIBCXX_3.4.5) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find_last_not_of(wchar_t const*, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find_last_not_of(wchar_t const*, unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find_last_not_of(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find_last_not_of(wchar_t, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find_first_not_of(wchar_t const*, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find_first_not_of(wchar_t const*, unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find_first_not_of(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find_first_not_of(wchar_t, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::at(unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::copy(wchar_t*, unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find(wchar_t const*, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find(wchar_t const*, unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::find(wchar_t, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::rfind(wchar_t const*, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::rfind(wchar_t const*, unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::rfind(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::rfind(wchar_t, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::substr(unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::compare(unsigned long, unsigned long, wchar_t const*) const(GLIBCXX_3.4)</p>

[ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::compare(unsigned long, unsigned long, wchar_t const*, unsigned long) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::compare(unsigned long, unsigned long, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::compare(unsigned long, unsigned long, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::M_check(unsigned long, char const*) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::M_limit(unsigned long, unsigned long) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::operator[] (unsigned long) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::S_construct(unsigned long, wchar_t, allocator<wchar_t> const&) (GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::M_replace_aux(unsigned long, unsigned long, unsigned long, wchar_t) (GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::M_replace_safe(unsigned long, unsigned long, wchar_t const*, unsigned long) (GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::at(unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::Rep::M_set_length_and_sharable(unsigned long)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::Rep::M_clone(allocator<wchar_t> const&, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::Rep::S_create(unsigned long, unsigned long, allocator<wchar_t> const&) (GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::erase(unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::append(wchar_t const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::append(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::append(unsigned long, wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::assign(wchar_t const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::assign(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]

<code>long, unsigned long, unsigned long, wchar_t)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::reserve(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::M_assign(wchar_t*, unsigned long, wchar_t)(GLIBCXX_3.4.5) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::M_mutate(unsigned long, unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::basic_string(wchar_t const*, unsigned long, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::basic_string(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::basic_string(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long, unsigned long, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::basic_string(unsigned long, wchar_t, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::basic_string(wchar_t const*, unsigned long, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::basic_string(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::basic_string(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&, unsigned long, unsigned long, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::basic_string(unsigned long, wchar_t, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::operator[](unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.60 Class `basic_stringstream<char, char_traits<char>, allocator<char>>`

12.1.60.1 Class data for `basic_stringstream<char, char_traits<char>, allocator<char>>`

The virtual table for the `std::basic_stringstream<char, std::char_traits<char>, std::allocator<char>>` class is described by [Table 12-47](#)

Table 12-47 Primary vtable for `basic_stringstream<char, char_traits<char>, allocator<char>>`

Base Offset	0
Virtual Base Offset	104
RTTI	typeinfo for <code>basic_stringstream<char, char_traits<char>, allocator<char>></code>

vfunc[0]:	basic_stringstream<char, char_traits<char>, allocator<char>>::~basic_stringstream()
vfunc[1]:	basic_stringstream<char, char_traits<char>, allocator<char>>::~basic_stringstream()

Table 12-48 Secondary vtable for basic_stringstream<char, char_traits<char>, allocator<char>>

Base Offset	-16
Virtual Base Offset	88
RTTI	typeinfo for basic_stringstream<char, char_traits<char>, allocator<char>>
vfunc[0]:	non-virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char>>::~basic_stringstream()
vfunc[1]:	non-virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char>>::~basic_stringstream()

Table 12-49 Secondary vtable for basic_stringstream<char, char_traits<char>, allocator<char>>

Base Offset	-104
Virtual Base Offset	-104
RTTI	typeinfo for basic_stringstream<char, char_traits<char>, allocator<char>>
vfunc[0]:	virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char>>::~basic_stringstream()
vfunc[1]:	virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char>>::~basic_stringstream()

The VTT for the std::basic_stringstream<char, std::char_traits<char>, std::allocator<char>> class is described by [Table 12-50](#)

Table 12-50 VTT for basic_stringstream<char, char_traits<char>, allocator<char>>

VTT Name	_ZTTSt18basic_stringstreamIcSt11char_traitsIcESaIcEE
Number of Entries	10

12.1.60.2 Interfaces for Class basic_stringstream<char, char_traits<char>, allocator<char>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_stringstream<char, std::char_traits<char>, std::allocator<char>> specified in [Table 12-51](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-51 libstdc++ - Class `basic_stringstream<char, char_traits<char>, allocator<char>>` Function Interfaces

non-virtual thunk to <code>basic_stringstream<char, char_traits<char>, allocator<char>>::~basic_stringstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to <code>basic_stringstream<char, char_traits<char>, allocator<char>>::~basic_stringstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to <code>basic_stringstream<char, char_traits<char>, allocator<char>>::~basic_stringstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to <code>basic_stringstream<char, char_traits<char>, allocator<char>>::~basic_stringstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]

12.1.61 Class `basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

12.1.61.1 Class data for `basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

The virtual table for the `std::basic_stringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>>` class is described by [Table 12-52](#)

Table 12-52 Primary vtable for `basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

Base Offset	0
Virtual Base Offset	104
RTTI	typeid for <code>basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>></code>
vfunc[0]:	<code>basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_stringstream()</code>
vfunc[1]:	<code>basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_stringstream()</code>

Table 12-53 Secondary vtable for `basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

Base Offset	-16
Virtual Base Offset	88
RTTI	typeid for <code>basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>></code>
vfunc[0]:	non-virtual thunk to <code>basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_stringstream()</code>
vfunc[1]:	non-virtual thunk to <code>basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>></code>

	allocator<wchar_t> >::~basic_stringstream()
--	--

Table 12-54 Secondary vtable for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

Base Offset	-104
Virtual Base Offset	-104
RTTI	typeid for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>
vfunc[0]:	virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringstream()
vfunc[1]:	virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringstream()

The VTT for the std::basic_stringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> class is described by [Table 12-55](#)

Table 12-55 VTT for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

VTT Name	_ZTTSt18basic_stringstreamIwSt11char_traitsIwESaIwEE
Number of Entries	10

12.1.61.2 Interfaces for Class basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_stringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> specified in [Table 12-56](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-56 libstdcxx - Class basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> Function Interfaces

non-virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]

12.1.62 Class `basic_istream<char, char_traits<char>, allocator<char>>`

12.1.62.1 Class data for `basic_istream<char, char_traits<char>, allocator<char>>`

The virtual table for the `std::basic_istream<char, std::char_traits<char>, std::allocator<char>>` class is described by [Table 12-57](#)

Table 12-57 Primary vtable for `basic_istream<char, char_traits<char>, allocator<char>>`

Base Offset	0
Virtual Base Offset	96
RTTI	typeinfo for <code>basic_istream<char, char_traits<char>, allocator<char>></code>
vfunc[0]:	<code>basic_istream<char, char_traits<char>, allocator<char>>::~~basic_istream()</code>
vfunc[1]:	<code>basic_istream<char, char_traits<char>, allocator<char>>::~~basic_istream()</code>

Table 12-58 Secondary vtable for `basic_istream<char, char_traits<char>, allocator<char>>`

Base Offset	-96
Virtual Base Offset	-96
RTTI	typeinfo for <code>basic_istream<char, char_traits<char>, allocator<char>></code>
vfunc[0]:	virtual thunk to <code>basic_istream<char, char_traits<char>, allocator<char>>::~~basic_istream()</code>
vfunc[1]:	virtual thunk to <code>basic_istream<char, char_traits<char>, allocator<char>>::~~basic_istream()</code>

The VTT for the `std::basic_istream<char, std::char_traits<char>, std::allocator<char>>` class is described by [Table 12-59](#)

Table 12-59 VTT for `basic_istream<char, char_traits<char>, allocator<char>>`

VTT Name	<code>_ZTTSt19basic_istreamIcSt11char_traitsIcESaIcEE</code>
Number of Entries	4

12.1.62.2 Interfaces for Class `basic_istream<char, char_traits<char>, allocator<char>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::basic_istream<char, std::char_traits<char>, std::allocator<char>>` specified in [Table 12-60](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-60 libstdcxx - Class basic_istream<char, char_traits<char>, allocator<char>> Function Interfaces

virtual thunk to basic_istream<char, char_traits<char>, allocator<char>>::~basic_istream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_istream<char, char_traits<char>, allocator<char>>::~basic_istream()(GLIBCXX_3.4) [CXXABI-1.86]

12.1.63 Class basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

12.1.63.1 Class data for basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

The virtual table for the std::basic_istream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> class is described by [Table 12-61](#)

Table 12-61 Primary vtable for basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

Base Offset	0
Virtual Base Offset	96
RTTI	typeid for basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>
vfunc[0]:	basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_istream()
vfunc[1]:	basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_istream()

Table 12-62 Secondary vtable for basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

Base Offset	-96
Virtual Base Offset	-96
RTTI	typeid for basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>
vfunc[0]:	virtual thunk to basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_istream()
vfunc[1]:	virtual thunk to basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_istream()

The VTT for the std::basic_istream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> class is described by [Table 12-63](#)

Table 12-63 VTT for basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

VTT Name	_ZTTSt19basic_istreamIwSt11char_traitsIwESaIwEE
Number of Entries	4

12.1.63.2 Interfaces for Class basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class `std::basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >` specified in [Table 12-64](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-64 libstdc++ - Class basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > Function Interfaces

virtual thunk to <code>basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_istream()(GLIBCXX_3.4)</code> [CXXABI-1.86]
virtual thunk to <code>basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_istream()(GLIBCXX_3.4)</code> [CXXABI-1.86]

12.1.64 Class basic_ostringstream<char, char_traits<char>, allocator<char> >

12.1.64.1 Class data for basic_ostringstream<char, char_traits<char>, allocator<char> >

The virtual table for the `std::basic_ostringstream<char, char_traits<char>, allocator<char> >` class is described by [Table 12-65](#)

Table 12-65 Primary vtable for basic_ostringstream<char, char_traits<char>, allocator<char> >

Base Offset	0
Virtual Base Offset	88
RTTI	typeinfo for <code>basic_ostringstream<char, char_traits<char>, allocator<char> ></code>
vfunc[0]:	<code>basic_ostringstream<char, char_traits<char>, allocator<char> >::~basic_ostringstream()</code>
vfunc[1]:	<code>basic_ostringstream<char, char_traits<char>, allocator<char> >::~basic_ostringstream()</code>

Table 12-66 Secondary vtable for basic_ostringstream<char, char_traits<char>, allocator<char> >

Base Offset	-88
Virtual Base Offset	-88
RTTI	typeinfo for <code>basic_ostringstream<char, char_traits<char>, allocator<char> ></code>
vfunc[0]:	virtual thunk to <code>basic_ostringstream<char, char_traits<char>, allocator<char> >::~basic_ostringstream()</code>
vfunc[1]:	virtual thunk to

	basic_ostringstream<char, char_traits<char>, allocator<char>>::~basic_ostringstream()
--	---

The VTT for the std::basic_ostringstream<char, std::char_traits<char>, std::allocator<char>> class is described by [Table 12-67](#)

Table 12-67 VTT for basic_ostringstream<char, char_traits<char>, allocator<char>>

VTT Name	_ZTTSt19basic_ostringstreamIcSt11char_traitsIcESaIcEE
Number of Entries	4

12.1.64.2 Interfaces for Class basic_ostringstream<char, char_traits<char>, allocator<char>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ostringstream<char, std::char_traits<char>, std::allocator<char>> specified in [Table 12-68](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-68 libstdc++ - Class basic_ostringstream<char, char_traits<char>, allocator<char>> Function Interfaces

virtual thunk to basic_ostringstream<char, char_traits<char>, allocator<char>>::~basic_ostringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ostringstream<char, char_traits<char>, allocator<char>>::~basic_ostringstream()(GLIBCXX_3.4) [CXXABI-1.86]

12.1.65 Class basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

12.1.65.1 Class data for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

The virtual table for the std::basic_ostringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> class is described by [Table 12-69](#)

Table 12-69 Primary vtable for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

Base Offset	0
Virtual Base Offset	88
RTTI	typeinfo for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>
vfunc[0]:	basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_ostringstream()
vfunc[1]:	basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~basic_ostringstream()

Table 12-70 Secondary vtable for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

Base Offset	-88
Virtual Base Offset	-88
RTTI	typeid for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >
vfunc[0]:	virtual thunk to basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_ostringstream()
vfunc[1]:	virtual thunk to basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_ostringstream()

The VTT for the std::basic_ostringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > class is described by [Table 12-71](#)

Table 12-71 VTT for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

VTT Name	_ZTTSt19basic_ostringstreamIwSt11char_traitsIwESaIwEE
Number of Entries	4

12.1.65.2 Interfaces for Class basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ostringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > specified in [Table 12-72](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-72 libstdcxx - Class basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > Function Interfaces

virtual thunk to basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_ostringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_ostringstream()(GLIBCXX_3.4) [CXXABI-1.86]

12.1.66 Class basic_stringbuf<char, char_traits<char>, allocator<char> >

12.1.66.1 Class data for basic_stringbuf<char, char_traits<char>, allocator<char> >

The virtual table for the std::basic_stringbuf<char, std::char_traits<char>, std::allocator<char> > class is described by [Table 12-73](#)

Table 12-73 Primary vtable for basic_stringbuf<char, char_traits<char>, allocator<char> >

Base Offset	0
-------------	---

Virtual Base Offset	0
RTTI	typeinfo for basic_stringbuf<char, char_traits<char>, allocator<char> >
vfunc[0]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::~~basic_stringbuf()
vfunc[1]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::~~basic_stringbuf()
vfunc[2]:	basic_streambuf<char, char_traits<char> >::imbue(locale const&)
vfunc[3]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::setbuf(char*, long)
vfunc[4]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::seekoff(long, _Ios_Seekdir, _Ios_Openmode)
vfunc[5]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_streambuf<char, char_traits<char> >::sync()
vfunc[7]:	basic_streambuf<char, char_traits<char> >::showmanyc()
vfunc[8]:	basic_streambuf<char, char_traits<char> >::xsgetn(char*, long)
vfunc[9]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::underflow()
vfunc[10]:	basic_streambuf<char, char_traits<char> >::uflow()
vfunc[11]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::pbackfail(int)
vfunc[12]:	basic_streambuf<char, char_traits<char> >::xsputn(char const*, long)
vfunc[13]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::overflow(int)

The Run Time Type Information for the std::basic_stringbuf<char, std::char_traits<char>, std::allocator<char> > class is described by [Table 12-74](#)

Table 12-74 typeinfo for basic_stringbuf<char, char_traits<char>, allocator<char> >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for basic_stringbuf<char, char_traits<char>, allocator<char> >

12.1.66.2 Interfaces for Class basic_stringbuf<char, char_traits<char>, allocator<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_stringbuf<char, std::char_traits<char>, std::allocator<char> > specified in [Table 12-75](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-75 libstdcxx - Class basic_stringbuf<char, char_traits<char>, allocator<char>> Function Interfaces

basic_stringbuf<char, char_traits<char>, allocator<char>>::setbuf(char*, long) (GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char>>::_M_sync(char*, unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char>>::seekoff(long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]

12.1.67 Class basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

12.1.67.1 Class data for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

The virtual table for the std::basic_stringbuf<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> class is described by [Table 12-76](#)

Table 12-76 Primary vtable for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>
vfunc[0]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~~basic_stringbuf()
vfunc[1]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::~~basic_stringbuf()
vfunc[2]:	basic_streambuf<wchar_t, char_traits<wchar_t>>::imbue(locale const&)
vfunc[3]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::setbuf(wchar_t*, long)
vfunc[4]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::seekoff(long, _Ios_Seekdir, _Ios_Openmode)
vfunc[5]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_streambuf<wchar_t, char_traits<wchar_t>>::sync()
vfunc[7]:	basic_streambuf<wchar_t, char_traits<wchar_t>>::showmanyc()

vfunc[8]:	basic_streambuf<wchar_t, char_traits<wchar_t>>::xsgetn(wchar_t*, long)
vfunc[9]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::underflow()
vfunc[10]:	basic_streambuf<wchar_t, char_traits<wchar_t>>::uflow()
vfunc[11]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::pbackfail(unsigned int)
vfunc[12]:	basic_streambuf<wchar_t, char_traits<wchar_t>>::xsputn(wchar_t const*, long)
vfunc[13]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::overflow(unsigned int)

The Run Time Type Information for the std::basic_stringbuf<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> class is described by [Table 12-77](#)

Table 12-77 typeid for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

12.1.67.2 Interfaces for Class basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_stringbuf<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> specified in [Table 12-78](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-78 libstdc++ - Class basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> Function Interfaces

basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::setbuf(wchar_t*, long)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::M_sync(wchar_t*, unsigned long, unsigned long)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>::seekoff(long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]

12.1.68 Class basic_iostream<char, char_traits<char>>

12.1.68.1 Class data for basic_iostream<char, char_traits<char>>

The virtual table for the std::basic_iostream<char, std::char_traits<char>> class is de-

scribed by [Table 12-79](#)

Table 12-79 Primary vtable for `basic_istream<char, char_traits<char>>`

Base Offset	0
Virtual Base Offset	24
RTTI	typeid for <code>basic_istream<char, char_traits<char>></code>
vfunc[0]:	<code>basic_istream<char, char_traits<char>>::~~basic_istream()</code>
vfunc[1]:	<code>basic_istream<char, char_traits<char>>::~~basic_istream()</code>

Table 12-80 Secondary vtable for `basic_istream<char, char_traits<char>>`

Base Offset	-16
Virtual Base Offset	8
RTTI	typeid for <code>basic_istream<char, char_traits<char>></code>
vfunc[0]:	non-virtual thunk to <code>basic_istream<char, char_traits<char>>::~~basic_istream()</code>
vfunc[1]:	non-virtual thunk to <code>basic_istream<char, char_traits<char>>::~~basic_istream()</code>

Table 12-81 Secondary vtable for `basic_istream<char, char_traits<char>>`

Base Offset	-24
Virtual Base Offset	-24
RTTI	typeid for <code>basic_istream<char, char_traits<char>></code>
vfunc[0]:	virtual thunk to <code>basic_istream<char, char_traits<char>>::~~basic_istream()</code>
vfunc[1]:	virtual thunk to <code>basic_istream<char, char_traits<char>>::~~basic_istream()</code>

The VTT for the `std::basic_istream<char, std::char_traits<char>>` class is described by [Table 12-82](#)

Table 12-82 VTT for `basic_istream<char, char_traits<char>>`

VTT Name	<code>_ZTTSD</code>
Number of Entries	7

12.1.68.2 Interfaces for Class `basic_istream<char, char_traits<char>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::basic_istream<char, std::char_traits<char>>` specified in [Table 12-83](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-83 `libstdcxx` - Class `basic_istream<char, char_traits<char>>` Function Interfaces

non-virtual thunk to <code>basic_istream<char, char_traits<char>>::~~basic_istream()</code> (GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to <code>basic_istream<char, char_traits<char>>::~~basic_istream()</code>

(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_istream<char, char_traits<char> >::~basic_istream() (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_istream<char, char_traits<char> >::~basic_istream() (GLIBCXX_3.4) [CXXABI-1.86]

12.1.69 Class basic_istream<wchar_t, char_traits<wchar_t> >

12.1.69.1 Class data for basic_istream<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_istream<wchar_t, std::char_traits<wchar_t> > class is described by [Table 12-84](#)

Table 12-84 Primary vtable for basic_istream<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	24
RTTI	typeinfo for basic_istream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()
vfunc[1]:	basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()

Table 12-85 Secondary vtable for basic_istream<wchar_t, char_traits<wchar_t> >

Base Offset	-16
Virtual Base Offset	8
RTTI	typeinfo for basic_istream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	non-virtual thunk to basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()
vfunc[1]:	non-virtual thunk to basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()

Table 12-86 Secondary vtable for basic_istream<wchar_t, char_traits<wchar_t> >

Base Offset	-24
Virtual Base Offset	-24
RTTI	typeinfo for basic_istream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	virtual thunk to basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()
vfunc[1]:	virtual thunk to basic_istream<wchar_t,

	char_traits<wchar_t> >::~basic_istream()
--	---

The VTT for the std::basic_istream<wchar_t, std::char_traits<wchar_t> > class is described by [Table 12-87](#)

Table 12-87 VTT for basic_istream<wchar_t, char_traits<wchar_t> >

VTT Name	_ZTTSt14basic_istreamIwSt11char_traitsIwEE
Number of Entries	7

12.1.69.2 Interfaces for Class basic_istream<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_istream<wchar_t, std::char_traits<wchar_t> > specified in [Table 12-88](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-88 libstdc++ - Class basic_istream<wchar_t, char_traits<wchar_t> > Function Interfaces

non-virtual thunk to basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()(GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()(GLIBCXX_3.4) [CXXABI-1.86]

12.1.70 Class basic_istream<char, char_traits<char> >

12.1.70.1 Class data for basic_istream<char, char_traits<char> >

The virtual table for the std::basic_istream<char, std::char_traits<char> > class is described by [Table 12-89](#)

Table 12-89 Primary vtable for basic_istream<char, char_traits<char> >

Base Offset	0
Virtual Base Offset	16
RTTI	typeinfo for basic_istream<char, char_traits<char> >
vfunc[0]:	basic_istream<char, char_traits<char> >::~basic_istream()
vfunc[1]:	basic_istream<char, char_traits<char> >::~basic_istream()

Table 12-90 Secondary vtable for basic_istream<char, char_traits<char> >

Base Offset	-16
Virtual Base Offset	-16
RTTI	typeinfo for basic_istream<char, char_traits<char> >

vfunc[0]:	virtual thunk to basic_istream<char, char_traits<char>>::~~basic_istream()
vfunc[1]:	virtual thunk to basic_istream<char, char_traits<char>>::~~basic_istream()

The VTT for the std::basic_istream<char, std::char_traits<char>> class is described by [Table 12-91](#)

Table 12-91 VTT for basic_istream<char, char_traits<char>>

VTT Name	_ZTTSi
Number of Entries	2

12.1.70.2 Interfaces for Class basic_istream<char, char_traits<char>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_istream<char, std::char_traits<char>> specified in [Table 12-92](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-92 libstdcxx - Class basic_istream<char, char_traits<char>> Function Interfaces

basic_istream<char, char_traits<char>>::get(char*, long)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char>>::get(char*, long, char)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char>>::read(char*, long)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char>>::seekg(long, _Ios_Seekdir)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char>>::ignore(long)(GLIBCXX_3.4.5) [ISOCXX]
basic_istream<char, char_traits<char>>::ignore(long, int)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char>>::getline(char*, long)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char>>::getline(char*, long, char)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char>>::readsomewhat(char*, long)(GLIBCXX_3.4) [ISOCXX]
virtual thunk to basic_istream<char, char_traits<char>>::~~basic_istream() (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_istream<char, char_traits<char>>::~~basic_istream() (GLIBCXX_3.4) [CXXABI-1.86]

12.1.71 Class basic_istream<wchar_t, char_traits<wchar_t>>

12.1.71.1 Class data for basic_istream<wchar_t, char_traits<wchar_t>>

The virtual table for the std::basic_istream<wchar_t, std::char_traits<wchar_t>> class is described by [Table 12-93](#)

Table 12-93 Primary vtable for basic_istream<wchar_t, char_traits<wchar_t>>

Base Offset	0
Virtual Base Offset	16
RTTI	typeid for basic_istream<wchar_t, char_traits<wchar_t>>
vfunc[0]:	basic_istream<wchar_t, char_traits<wchar_t>>::~~basic_istream()
vfunc[1]:	basic_istream<wchar_t, char_traits<wchar_t>>::~~basic_istream()

Table 12-94 Secondary vtable for basic_istream<wchar_t, char_traits<wchar_t>>

Base Offset	-16
Virtual Base Offset	-16
RTTI	typeid for basic_istream<wchar_t, char_traits<wchar_t>>
vfunc[0]:	virtual thunk to basic_istream<wchar_t, char_traits<wchar_t>>::~~basic_istream()
vfunc[1]:	virtual thunk to basic_istream<wchar_t, char_traits<wchar_t>>::~~basic_istream()

The VTT for the `std::basic_istream<wchar_t, std::char_traits<wchar_t>>` class is described by [Table 12-95](#)

Table 12-95 VTT for basic_istream<wchar_t, char_traits<wchar_t>>

VTT Name	<code>_ZTTSt13basic_istreamIwSt11char_traitsIwEE</code>
Number of Entries	2

12.1.71.2 Interfaces for Class `basic_istream<wchar_t, char_traits<wchar_t>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::basic_istream<wchar_t, std::char_traits<wchar_t>>` specified in [Table 12-96](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-96 libstdc++ - Class `basic_istream<wchar_t, char_traits<wchar_t>>` Function Interfaces

<code>basic_istream<wchar_t, char_traits<wchar_t>>::get(wchar_t*, long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream<wchar_t, char_traits<wchar_t>>::get(wchar_t*, long, wchar_t)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream<wchar_t, char_traits<wchar_t>>::read(wchar_t*, long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream<wchar_t, char_traits<wchar_t>>::seekg(long, _Ios_Seekdir)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream<wchar_t, char_traits<wchar_t>>::ignore(long)</code> (GLIBCXX_3.4.5) [ISOCXX]
<code>basic_istream<wchar_t, char_traits<wchar_t>>::ignore(long, unsigned int)</code>

(GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t>>::getline(wchar_t*, long) (GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t>>::getline(wchar_t*, long, wchar_t) (GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t>>::readsome(wchar_t*, long) (GLIBCXX_3.4) [ISOCXX]
virtual thunk to basic_istream<wchar_t, char_traits<wchar_t>>::~basic_istream() (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_istream<wchar_t, char_traits<wchar_t>>::~basic_istream() (GLIBCXX_3.4) [CXXABI-1.86]

12.1.72 Class istreambuf_iterator<wchar_t, char_traits<wchar_t>>

12.1.72.1 Interfaces for Class istreambuf_iterator<wchar_t, char_traits<wchar_t>>

No external methods are defined for libstdc++ - Class std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>> in this part of the specification. See also the generic specification.

12.1.73 Class istreambuf_iterator<char, char_traits<char>>

12.1.73.1 Interfaces for Class istreambuf_iterator<char, char_traits<char>>

No external methods are defined for libstdc++ - Class std::istreambuf_iterator<char, std::char_traits<char>> in this part of the specification. See also the generic specification.

12.1.74 Class basic_ostream<char, char_traits<char>>

12.1.74.1 Class data for basic_ostream<char, char_traits<char>>

The virtual table for the std::basic_ostream<char, std::char_traits<char>> class is described by [Table 12-97](#)

Table 12-97 Primary vtable for basic_ostream<char, char_traits<char>>

Base Offset	0
Virtual Base Offset	8
RTTI	typeid for basic_ostream<char, char_traits<char>>
vfunc[0]:	basic_ostream<char, char_traits<char>>::~basic_ostream()
vfunc[1]:	basic_ostream<char, char_traits<char>>::~basic_ostream()

Table 12-98 Secondary vtable for basic_ostream<char, char_traits<char>>

Base Offset	-8
Virtual Base Offset	-8

RTTI	typeinfo for basic_ostream<char, char_traits<char> >
vfunc[0]:	virtual thunk to basic_ostream<char, char_traits<char> >::~basic_ostream()
vfunc[1]:	virtual thunk to basic_ostream<char, char_traits<char> >::~basic_ostream()

The VTT for the std::basic_ostream<char, std::char_traits<char> > class is described by [Table 12-99](#)

Table 12-99 VTT for basic_ostream<char, char_traits<char> >

VTT Name	_ZTTSo
Number of Entries	2

12.1.74.2 Interfaces for Class basic_ostream<char, char_traits<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ostream<char, std::char_traits<char> > specified in [Table 12-100](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-100 libstdcxx - Class basic_ostream<char, char_traits<char> > Function Interfaces

basic_ostream<char, char_traits<char> >::seekp(long, _Ios_Seekdir) (GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::write(char const*, long)(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::_M_write(char const*, long) (GLIBCXX_3.4) [ISOCXX]
virtual thunk to basic_ostream<char, char_traits<char> >::~basic_ostream() (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ostream<char, char_traits<char> >::~basic_ostream() (GLIBCXX_3.4) [CXXABI-1.86]

12.1.75 Class basic_ostream<wchar_t, char_traits<wchar_t> >

12.1.75.1 Class data for basic_ostream<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_ostream<wchar_t, std::char_traits<wchar_t> > class is described by [Table 12-101](#)

Table 12-101 Primary vtable for basic_ostream<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	8
RTTI	typeinfo for basic_ostream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_ostream<wchar_t, char_traits<wchar_t> >::~basic_ostream()
vfunc[1]:	basic_ostream<wchar_t, char_traits<wchar_t> >::~basic_ostream()

Table 12-102 Secondary vtable for basic_ostream<wchar_t, char_traits<wchar_t>>

Base Offset	-8
Virtual Base Offset	-8
RTTI	typeid for basic_ostream<wchar_t, char_traits<wchar_t>>
vfunc[0]:	virtual thunk to basic_ostream<wchar_t, char_traits<wchar_t>>::~basic_ostream()
vfunc[1]:	virtual thunk to basic_ostream<wchar_t, char_traits<wchar_t>>::~basic_ostream()

The VTT for the std::basic_ostream<wchar_t, std::char_traits<wchar_t>> class is described by [Table 12-103](#)

Table 12-103 VTT for basic_ostream<wchar_t, char_traits<wchar_t>>

VTT Name	_ZTTSt13basic_ostreamIwSt11char_traitsIwEE
Number of Entries	2

12.1.75.2 Interfaces for Class basic_ostream<wchar_t, char_traits<wchar_t>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ostream<wchar_t, std::char_traits<wchar_t>> specified in [Table 12-104](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-104 libstdc++ - Class basic_ostream<wchar_t, char_traits<wchar_t>> Function Interfaces

basic_ostream<wchar_t, char_traits<wchar_t>>::seekp(long, _Ios_Seekdir) (GLIBCXX_3.4) [ISOCXX]
basic_ostream<wchar_t, char_traits<wchar_t>>::write(wchar_t const*, long) (GLIBCXX_3.4) [ISOCXX]
virtual thunk to basic_ostream<wchar_t, char_traits<wchar_t>>::~basic_ostream() (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ostream<wchar_t, char_traits<wchar_t>>::~basic_ostream() (GLIBCXX_3.4) [CXXABI-1.86]

12.1.76 Class basic_fstream<char, char_traits<char>>

12.1.76.1 Class data for basic_fstream<char, char_traits<char>>

The virtual table for the std::basic_fstream<char, std::char_traits<char>> class is described by [Table 12-105](#)

Table 12-105 Primary vtable for basic_fstream<char, char_traits<char>>

Base Offset	0
Virtual Base Offset	264
RTTI	typeid for basic_fstream<char, char_traits<char>>
vfunc[0]:	basic_fstream<char, char_traits<char>>

	>::~~basic_fstream()
vfunc[1]:	basic_fstream<char, char_traits<char>> >::~~basic_fstream()

Table 12-106 Secondary vtable for basic_fstream<char, char_traits<char>>

Base Offset	-16
Virtual Base Offset	248
RTTI	typeid for basic_fstream<char, char_traits<char>>
vfunc[0]:	non-virtual thunk to basic_fstream<char, char_traits<char>>::~~basic_fstream()
vfunc[1]:	non-virtual thunk to basic_fstream<char, char_traits<char>>::~~basic_fstream()

Table 12-107 Secondary vtable for basic_fstream<char, char_traits<char>>

Base Offset	-264
Virtual Base Offset	-264
RTTI	typeid for basic_fstream<char, char_traits<char>>
vfunc[0]:	virtual thunk to basic_fstream<char, char_traits<char>>::~~basic_fstream()
vfunc[1]:	virtual thunk to basic_fstream<char, char_traits<char>>::~~basic_fstream()

The VTT for the std::basic_fstream<char, std::char_traits<char>> class is described by [Table 12-108](#)

Table 12-108 VTT for basic_fstream<char, char_traits<char>>

VTT Name	_ZTTSt13basic_fstreamIcSt11char_traitsEE
Number of Entries	10

12.1.76.2 Interfaces for Class basic_fstream<char, char_traits<char>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_fstream<char, std::char_traits<char>> specified in [Table 12-109](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-109 libstdc++ - Class basic_fstream<char, char_traits<char>> Function Interfaces

non-virtual thunk to basic_fstream<char, char_traits<char>>::~~basic_fstream() (GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to basic_fstream<char, char_traits<char>>::~~basic_fstream() (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_fstream<char, char_traits<char>>::~~basic_fstream() (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_fstream<char, char_traits<char>>::~~basic_fstream() (GLIBCXX_3.4) [CXXABI-1.86]

12.1.77 Class `basic_fstream<wchar_t, char_traits<wchar_t>>`

12.1.77.1 Class data for `basic_fstream<wchar_t, char_traits<wchar_t>>`

The virtual table for the `std::basic_fstream<wchar_t, std::char_traits<wchar_t>>` class is described by [Table 12-110](#)

Table 12-110 Primary vtable for `basic_fstream<wchar_t, char_traits<wchar_t>>`

Base Offset	0
Virtual Base Offset	264
RTTI	typeid for <code>basic_fstream<wchar_t, char_traits<wchar_t>></code>
vfunc[0]:	<code>basic_fstream<wchar_t, char_traits<wchar_t>>::~~basic_fstream()</code>
vfunc[1]:	<code>basic_fstream<wchar_t, char_traits<wchar_t>>::~~basic_fstream()</code>

Table 12-111 Secondary vtable for `basic_fstream<wchar_t, char_traits<wchar_t>>`

Base Offset	-16
Virtual Base Offset	248
RTTI	typeid for <code>basic_fstream<wchar_t, char_traits<wchar_t>></code>
vfunc[0]:	non-virtual thunk to <code>basic_fstream<wchar_t, char_traits<wchar_t>>::~~basic_fstream()</code>
vfunc[1]:	non-virtual thunk to <code>basic_fstream<wchar_t, char_traits<wchar_t>>::~~basic_fstream()</code>

Table 12-112 Secondary vtable for `basic_fstream<wchar_t, char_traits<wchar_t>>`

Base Offset	-264
Virtual Base Offset	-264
RTTI	typeid for <code>basic_fstream<wchar_t, char_traits<wchar_t>></code>
vfunc[0]:	virtual thunk to <code>basic_fstream<wchar_t, char_traits<wchar_t>>::~~basic_fstream()</code>
vfunc[1]:	virtual thunk to <code>basic_fstream<wchar_t, char_traits<wchar_t>>::~~basic_fstream()</code>

The VTT for the `std::basic_fstream<wchar_t, std::char_traits<wchar_t>>` class is described by [Table 12-113](#)

Table 12-113 VTT for basic_fstream<wchar_t, char_traits<wchar_t>>

VTT Name	_ZTTSt13basic_fstreamIwSt11char_traitsIwEE
Number of Entries	10

12.1.77.2 Interfaces for Class basic_fstream<wchar_t, char_traits<wchar_t>>

An LSB conforming implementation shall provide the architecture specific methods for Class `std::basic_fstream<wchar_t, std::char_traits<wchar_t>>` specified in [Table 12-114](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-114 libstdc++ - Class basic_fstream<wchar_t, char_traits<wchar_t>> Function Interfaces

non-virtual thunk to <code>basic_fstream<wchar_t, char_traits<wchar_t>>::~basic_fstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to <code>basic_fstream<wchar_t, char_traits<wchar_t>>::~basic_fstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to <code>basic_fstream<wchar_t, char_traits<wchar_t>>::~basic_fstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]

12.1.78 Class basic_ifstream<char, char_traits<char>>

12.1.78.1 Class data for basic_ifstream<char, char_traits<char>>

The virtual table for the `std::basic_ifstream<char, std::char_traits<char>>` class is described by [Table 12-115](#)

Table 12-115 Primary vtable for basic_ifstream<char, char_traits<char>>

Base Offset	0
Virtual Base Offset	256
RTTI	<code>typeinfo for basic_ifstream<char, char_traits<char>></code>
<code>vfunc[0]:</code>	<code>basic_ifstream<char, char_traits<char>>::~basic_ifstream()</code>
<code>vfunc[1]:</code>	<code>basic_ifstream<char, char_traits<char>>::~basic_ifstream()</code>

Table 12-116 Secondary vtable for basic_ifstream<char, char_traits<char>>

Base Offset	-256
Virtual Base Offset	-256
RTTI	<code>typeinfo for basic_ifstream<char, char_traits<char>></code>
<code>vfunc[0]:</code>	virtual thunk to <code>basic_ifstream<char, char_traits<char>>::~basic_ifstream()</code>
<code>vfunc[1]:</code>	virtual thunk to <code>basic_ifstream<char, char_traits<char>>::~basic_ifstream()</code>

The VTT for the `std::basic_ifstream<char, std::char_traits<char>>` class is described by [Table 12-117](#)

Table 12-117 VTT for basic_ifstream<char, char_traits<char> >

VTT Name	_ZTTSt14basic_ifstreamIcSt11char_traitsIcEE
Number of Entries	4

12.1.78.2 Interfaces for Class basic_ifstream<char, char_traits<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ifstream<char, std::char_traits<char> > specified in [Table 12-118](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-118 libstdcxx - Class basic_ifstream<char, char_traits<char> > Function Interfaces

virtual thunk to basic_ifstream<char, char_traits<char> >::~basic_ifstream() (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ifstream<char, char_traits<char> >::~basic_ifstream() (GLIBCXX_3.4) [CXXABI-1.86]

12.1.79 Class basic_ifstream<wchar_t, char_traits<wchar_t> >

12.1.79.1 Class data for basic_ifstream<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_ifstream<wchar_t, std::char_traits<wchar_t> > class is described by [Table 12-119](#)

Table 12-119 Primary vtable for basic_ifstream<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	256
RTTI	typeid for basic_ifstream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_ifstream<wchar_t, char_traits<wchar_t> >::~basic_ifstream()
vfunc[1]:	basic_ifstream<wchar_t, char_traits<wchar_t> >::~basic_ifstream()

Table 12-120 Secondary vtable for basic_ifstream<wchar_t, char_traits<wchar_t> >

Base Offset	-256
Virtual Base Offset	-256
RTTI	typeid for basic_ifstream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	virtual thunk to basic_ifstream<wchar_t, char_traits<wchar_t> >::~basic_ifstream()
vfunc[1]:	virtual thunk to basic_ifstream<wchar_t, char_traits<wchar_t> >::~basic_ifstream()

The VTT for the std::basic_ifstream<wchar_t, std::char_traits<wchar_t> > class is de-

scribed by [Table 12-121](#)

Table 12-121 VTT for basic_ifstream<wchar_t, char_traits<wchar_t>>

VTT Name	_ZTTSt14basic_ifstreamIwSt11char_traitsIwEE
Number of Entries	4

12.1.79.2 Interfaces for Class basic_ifstream<wchar_t, char_traits<wchar_t>>

An LSB conforming implementation shall provide the architecture specific methods for Class `std::basic_ifstream<wchar_t, std::char_traits<wchar_t>>` specified in [Table 12-122](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-122 libstdc++ - Class basic_ifstream<wchar_t, char_traits<wchar_t>> Function Interfaces

virtual thunk to <code>basic_ifstream<wchar_t, char_traits<wchar_t>>::~basic_ifstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to <code>basic_ifstream<wchar_t, char_traits<wchar_t>>::~basic_ifstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]

12.1.80 Class basic_ofstream<char, char_traits<char>>

12.1.80.1 Class data for basic_ofstream<char, char_traits<char>>

The virtual table for the `std::basic_ofstream<char, std::char_traits<char>>` class is described by [Table 12-123](#)

Table 12-123 Primary vtable for basic_ofstream<char, char_traits<char>>

Base Offset	0
Virtual Base Offset	248
RTTI	typeinfo for <code>basic_ofstream<char, char_traits<char>></code>
vfunc[0]:	<code>basic_ofstream<char, char_traits<char>>::~basic_ofstream()</code>
vfunc[1]:	<code>basic_ofstream<char, char_traits<char>>::~basic_ofstream()</code>

Table 12-124 Secondary vtable for basic_ofstream<char, char_traits<char>>

Base Offset	-248
Virtual Base Offset	-248
RTTI	typeinfo for <code>basic_ofstream<char, char_traits<char>></code>
vfunc[0]:	virtual thunk to <code>basic_ofstream<char, char_traits<char>>::~basic_ofstream()</code>
vfunc[1]:	virtual thunk to <code>basic_ofstream<char, char_traits<char>>::~basic_ofstream()</code>

The VTT for the `std::basic_ofstream<char, std::char_traits<char>>` class is described by [Table 12-125](#)

Table 12-125 VTT for basic_ofstream<char, char_traits<char> >

VTT Name	_ZTTSt14basic_ofstreamIcSt11char_traitsIcEE
Number of Entries	4

12.1.80.2 Interfaces for Class basic_ofstream<char, char_traits<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ofstream<char, std::char_traits<char> > specified in [Table 12-126](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-126 libstdcxx - Class basic_ofstream<char, char_traits<char> > Function Interfaces

virtual thunk to basic_ofstream<char, char_traits<char> >::~basic_ofstream() (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ofstream<char, char_traits<char> >::~basic_ofstream() (GLIBCXX_3.4) [CXXABI-1.86]

12.1.81 Class basic_ofstream<wchar_t, char_traits<wchar_t> >

12.1.81.1 Class data for basic_ofstream<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_ofstream<wchar_t, std::char_traits<wchar_t> > class is described by [Table 12-127](#)

Table 12-127 Primary vtable for basic_ofstream<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	248
RTTI	typeid for basic_ofstream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()
vfunc[1]:	basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()

Table 12-128 Secondary vtable for basic_ofstream<wchar_t, char_traits<wchar_t> >

Base Offset	-248
Virtual Base Offset	-248
RTTI	typeid for basic_ofstream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	virtual thunk to basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()
vfunc[1]:	virtual thunk to basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()

The VTT for the `std::basic_ofstream<wchar_t, std::char_traits<wchar_t> >` class is described by [Table 12-129](#)

Table 12-129 VTT for `basic_ofstream<wchar_t, char_traits<wchar_t> >`

VTT Name	<code>_ZTTSt14basic_ofstreamIwSt11char_traitsIwEE</code>
Number of Entries	4

12.1.81.2 Interfaces for Class `basic_ofstream<wchar_t, char_traits<wchar_t> >`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::basic_ofstream<wchar_t, std::char_traits<wchar_t> >` specified in [Table 12-130](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-130 `libstdcxx` - Class `basic_ofstream<wchar_t, char_traits<wchar_t> >` Function Interfaces

virtual thunk to <code>basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to <code>basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]

12.1.82 Class `basic_streambuf<char, char_traits<char> >`

12.1.82.1 Class data for `basic_streambuf<char, char_traits<char> >`

The virtual table for the `std::basic_streambuf<char, std::char_traits<char> >` class is described by [Table 12-131](#)

Table 12-131 Primary vtable for `basic_streambuf<char, char_traits<char> >`

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeinfo for basic_streambuf<char, char_traits<char> ></code>
<code>vfunc[0]:</code>	<code>basic_streambuf<char, char_traits<char> >::~basic_streambuf()</code>
<code>vfunc[1]:</code>	<code>basic_streambuf<char, char_traits<char> >::~basic_streambuf()</code>
<code>vfunc[2]:</code>	<code>basic_streambuf<char, char_traits<char> >::imbue(locale const&)</code>
<code>vfunc[3]:</code>	<code>basic_streambuf<char, char_traits<char> >::setbuf(char*, long)</code>
<code>vfunc[4]:</code>	<code>basic_streambuf<char, char_traits<char> >::seekoff(long, _Ios_Seekdir, _Ios_Openmode)</code>
<code>vfunc[5]:</code>	<code>basic_streambuf<char, char_traits<char> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)</code>
<code>vfunc[6]:</code>	<code>basic_streambuf<char, char_traits<char> >::sync()</code>
<code>vfunc[7]:</code>	<code>basic_streambuf<char, char_traits<char> >::showmanyc()</code>

vfunc[8]:	basic_streambuf<char, char_traits<char>>::xsgetn(char*, long)
vfunc[9]:	basic_streambuf<char, char_traits<char>>::underflow()
vfunc[10]:	basic_streambuf<char, char_traits<char>>::uflow()
vfunc[11]:	basic_streambuf<char, char_traits<char>>::pbackfail(int)
vfunc[12]:	basic_streambuf<char, char_traits<char>>::xsputn(char const*, long)
vfunc[13]:	basic_streambuf<char, char_traits<char>>::overflow(int)

The Run Time Type Information for the std::basic_streambuf<char, std::char_traits<char>> class is described by [Table 12-132](#)

Table 12-132 typeid for basic_streambuf<char, char_traits<char>>

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeid name for basic_streambuf<char, char_traits<char>> >

12.1.82.2 Interfaces for Class basic_streambuf<char, char_traits<char>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_streambuf<char, std::char_traits<char>> specified in [Table 12-133](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-133 libstdc++ - Class basic_streambuf<char, char_traits<char>> > Function Interfaces

basic_streambuf<char, char_traits<char>>::pubseekoff(long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char>>::sgetn(char*, long)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char>>::sputn(char const*, long)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char>>::setbuf(char*, long)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char>>::xsgetn(char*, long)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char>>::xsputn(char const*, long)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char>>::seekoff(long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char>>::pubsetbuf(char*, long)(GLIBCXX_3.4) [ISOCXX]

12.1.83 Class `basic_streambuf<wchar_t, char_traits<wchar_t>>`

12.1.83.1 Class data for `basic_streambuf<wchar_t, char_traits<wchar_t>>`

The virtual table for the `std::basic_streambuf<wchar_t, std::char_traits<wchar_t>>` class is described by [Table 12-134](#)

Table 12-134 Primary vtable for `basic_streambuf<wchar_t, char_traits<wchar_t>>`

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeinfo for basic_streambuf<wchar_t, char_traits<wchar_t>></code>
<code>vfunc[0]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::~~basic_streambuf()</code>
<code>vfunc[1]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::~~basic_streambuf()</code>
<code>vfunc[2]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::imbue(locale const&)</code>
<code>vfunc[3]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::setbuf(wchar_t*, long)</code>
<code>vfunc[4]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::seekoff(long, _Ios_Seekdir, _Ios_Openmode)</code>
<code>vfunc[5]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::seekpos(fpos<__mbstate_t>, _Ios_Openmode)</code>
<code>vfunc[6]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::sync()</code>
<code>vfunc[7]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::showmanyc()</code>
<code>vfunc[8]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::xsgetn(wchar_t*, long)</code>
<code>vfunc[9]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::underflow()</code>
<code>vfunc[10]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::uflow()</code>
<code>vfunc[11]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::pbackfail(unsigned int)</code>
<code>vfunc[12]:</code>	<code>basic_streambuf<wchar_t, char_traits<wchar_t>>::xsputn(wchar_t const*, long)</code>
<code>vfunc[13]:</code>	<code>basic_streambuf<wchar_t,</code>

	char_traits<wchar_t> >::overflow(unsigned int)
--	---

The Run Time Type Information for the std::basic_streambuf<wchar_t, std::char_traits<wchar_t> > class is described by [Table 12-135](#)

Table 12-135 typeid for basic_streambuf<wchar_t, char_traits<wchar_t> >

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeid name for basic_streambuf<wchar_t, char_traits<wchar_t> >

12.1.83.2 Interfaces for Class basic_streambuf<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_streambuf<wchar_t, std::char_traits<wchar_t> > specified in [Table 12-136](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-136 libstdc++ - Class basic_streambuf<wchar_t, char_traits<wchar_t> > Function Interfaces

basic_streambuf<wchar_t, char_traits<wchar_t> >::pubseekoff(long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::sgetn(wchar_t*, long)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::sputn(wchar_t const*, long)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::setbuf(wchar_t*, long)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::xsgetn(wchar_t*, long)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::xsputn(wchar_t const*, long)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::seekoff(long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::pubsetbuf(wchar_t*, long)(GLIBCXX_3.4) [ISOCXX]

12.1.84 Class basic_filebuf<char, char_traits<char> >

12.1.84.1 Class data for basic_filebuf<char, char_traits<char> >

The virtual table for the std::basic_filebuf<char, std::char_traits<char> > class is described by [Table 12-137](#)

Table 12-137 Primary vtable for basic_filebuf<char, char_traits<char> >

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for basic_filebuf<char, char_traits<char> >
vfunc[0]:	basic_filebuf<char, char_traits<char> > >::~basic_filebuf()

vfunc[1]:	basic_filebuf<char, char_traits<char>>::~~basic_filebuf()
vfunc[2]:	basic_filebuf<char, char_traits<char>>::~imbue(locale const&)
vfunc[3]:	basic_filebuf<char, char_traits<char>>::~setbuf(char*, long)
vfunc[4]:	basic_filebuf<char, char_traits<char>>::~seekoff(long, _Ios_Seekdir, _Ios_Openmode)
vfunc[5]:	basic_filebuf<char, char_traits<char>>::~seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_filebuf<char, char_traits<char>>::~sync()
vfunc[7]:	basic_filebuf<char, char_traits<char>>::~showmanyc()
vfunc[8]:	basic_filebuf<char, char_traits<char>>::~xsgetn(char*, long)
vfunc[9]:	basic_filebuf<char, char_traits<char>>::~underflow()
vfunc[10]:	basic_streambuf<char, char_traits<char>>::~uflow()
vfunc[11]:	basic_filebuf<char, char_traits<char>>::~pbackfail(int)
vfunc[12]:	basic_filebuf<char, char_traits<char>>::~xsputn(char const*, long)
vfunc[13]:	basic_filebuf<char, char_traits<char>>::~overflow(int)

The Run Time Type Information for the `std::basic_filebuf<char, std::char_traits<char>>` class is described by [Table 12-138](#)

Table 12-138 typeid for `basic_filebuf<char, char_traits<char>>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>basic_filebuf<char, char_traits<char>></code>

12.1.84.2 Interfaces for Class `basic_filebuf<char, char_traits<char>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::basic_filebuf<char, std::char_traits<char>>` specified in [Table 12-139](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-139 libstdc++ - Class `basic_filebuf<char, char_traits<char>>` Function Interfaces

<code>basic_filebuf<char, char_traits<char>>::_M_set_buffer(long)(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf<char, char_traits<char>>::_M_convert_to_external(char*, long)(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf<char, char_traits<char>>::setbuf(char*, long)(GLIBCXX_3.4)</code> [ISOCXX]

basic_filebuf<char, char_traits<char> >::xsgetn(char*, long)(GLIBCXX_3.4) [ISO CXX]
basic_filebuf<char, char_traits<char> >::xsputn(char const*, long)(GLIBCXX_3.4) [ISO CXX]
basic_filebuf<char, char_traits<char> >::_M_seek(long, _Ios_Seekdir, __mbstate_t)(GLIBCXX_3.4) [ISO CXX]
basic_filebuf<char, char_traits<char> >::seekoff(long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISO CXX]

12.1.85 Class basic_filebuf<wchar_t, char_traits<wchar_t> >

12.1.85.1 Class data for basic_filebuf<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_filebuf<wchar_t, std::char_traits<wchar_t> > class is described by [Table 12-140](#)

Table 12-140 Primary vtable for basic_filebuf<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for basic_filebuf<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::~~basic_filebuf()
vfunc[1]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::~~basic_filebuf()
vfunc[2]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::imbue(locale const&)
vfunc[3]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::setbuf(wchar_t*, long)
vfunc[4]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::seekoff(long, _Ios_Seekdir, _Ios_Openmode)
vfunc[5]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::sync()
vfunc[7]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::showmanyc()
vfunc[8]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::xsgetn(wchar_t*, long)
vfunc[9]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::underflow()
vfunc[10]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::uflow()
vfunc[11]:	basic_filebuf<wchar_t,

	char_traits<wchar_t> >::pbackfail(unsigned int)
vfunc[12]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::xsputn(wchar_t const*, long)
vfunc[13]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::overflow(unsigned int)

The Run Time Type Information for the `std::basic_filebuf<wchar_t, std::char_traits<wchar_t> >` class is described by [Table 12-141](#)

Table 12-141 typeid for `basic_filebuf<wchar_t, char_traits<wchar_t> >`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>basic_filebuf<wchar_t, char_traits<wchar_t> ></code>

12.1.85.2 Interfaces for Class `basic_filebuf<wchar_t, char_traits<wchar_t> >`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::basic_filebuf<wchar_t, std::char_traits<wchar_t> >` specified in [Table 12-142](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-142 `libstdcxx` - Class `basic_filebuf<wchar_t, char_traits<wchar_t> >` Function Interfaces

<code>basic_filebuf<wchar_t, char_traits<wchar_t> >::_M_set_buffer(long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_filebuf<wchar_t, char_traits<wchar_t> >::_M_convert_to_external(wchar_t*, long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_filebuf<wchar_t, char_traits<wchar_t> >::setbuf(wchar_t*, long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_filebuf<wchar_t, char_traits<wchar_t> >::xsgetn(wchar_t*, long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_filebuf<wchar_t, char_traits<wchar_t> >::xsputn(wchar_t const*, long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_filebuf<wchar_t, char_traits<wchar_t> >::_M_seek(long, _Ios_Seekdir, __mbstate_t)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_filebuf<wchar_t, char_traits<wchar_t> >::seekoff(long, _Ios_Seekdir, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ostream<wchar_t, char_traits<wchar_t> >::_M_write(wchar_t const*, long)</code> (GLIBCXX_3.4) [ISOCXX]
virtual thunk to <code>basic_fstream<wchar_t, char_traits<wchar_t> >::~basic_fstream()</code> (GLIBCXX_3.4) [CXXABI-1.86]

12.1.86 Class `ios_base`

12.1.86.1 Class data for `ios_base`

The virtual table for the `std::ios_base` class is described in the generic part of this specification.

The Run Time Type Information for the `std::ios_base` class is described by [Table 12-143](#)

Table 12-143 typeinfo for `ios_base`

Base Vtable	vtable for <code>__cxxabiv1::__class_type_info</code>
Name	typeinfo name for <code>ios_base</code>

12.1.86.2 Interfaces for Class `ios_base`

No external methods are defined for `libstdc++` - Class `std::ios_base` in this part of the specification. See also the generic specification.

12.1.87 Class `basic_ios<char, char_traits<char>>`

12.1.87.1 Class data for `basic_ios<char, char_traits<char>>`

The virtual table for the `std::basic_ios<char, std::char_traits<char>>` class is described in the generic part of this specification.

12.1.87.2 Interfaces for Class `basic_ios<char, char_traits<char>>`

No external methods are defined for `libstdc++` - Class `std::basic_ios<char, std::char_traits<char>>` in this part of the specification. See also the generic specification.

12.1.88 Class `basic_ios<wchar_t, char_traits<wchar_t>>`

12.1.88.1 Class data for `basic_ios<wchar_t, char_traits<wchar_t>>`

The virtual table for the `std::basic_ios<wchar_t, std::char_traits<wchar_t>>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::basic_ios<wchar_t, std::char_traits<wchar_t>>` class is described by [Table 12-144](#)

Table 12-144 typeinfo for `basic_ios<wchar_t, char_traits<wchar_t>>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>	1026
Name	typeinfo name for <code>basic_ios<wchar_t, char_traits<wchar_t>></code>	
flags:	8	
basetype:	typeinfo for <code>ios_base</code>	

12.1.88.2 Interfaces for Class `basic_ios<wchar_t, char_traits<wchar_t>>`

No external methods are defined for `libstdc++` - Class `std::basic_ios<wchar_t, std::char_traits<wchar_t>>` in this part of the specification. See also the generic specification.

12.1.89 Class `ios_base::failure`

12.1.89.1 Class data for `ios_base::failure`

The virtual table for the `std::ios_base::failure` class is described in the generic part of this specification.

The Run Time Type Information for the `std::ios_base::failure` class is described by [Table 12-145](#)

Table 12-145 typeinfo for `ios_base::failure`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>ios_base::failure</code>

12.1.89.2 Interfaces for Class `ios_base::failure`

No external methods are defined for `libstdcxx` - Class `std::ios_base::failure` in this part of the specification. See also the generic specification.

12.1.90 Class `__timepunct<char>`

12.1.90.1 Class data for `__timepunct<char>`

The virtual table for the `std::__timepunct<char>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::__timepunct<char>` class is described by [Table 12-146](#)

Table 12-146 typeinfo for `__timepunct<char>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>__timepunct<char></code>

12.1.90.2 Interfaces for Class `__timepunct<char>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::__timepunct<char>` specified in [Table 12-147](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-147 `libstdcxx` - Class `__timepunct<char>` Function Interfaces

<code>__timepunct<char>::M_put(char*, unsigned long, char const*, tm const*)</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct<char>::__timepunct(__locale_struct*, char const*, unsigned long)</code> <code>(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct<char>::__timepunct(__timepunct_cache<char>*, unsigned long)</code> <code>(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct<char>::__timepunct(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct<char>::timepunct(__locale_struct*, char const*, unsigned long)</code> <code>(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct<char>::timepunct(__timepunct_cache<char>*, unsigned long)</code> <code>(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct<char>::timepunct(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.91 Class `__timepunct<wchar_t>`

12.1.91.1 Class data for `__timepunct<wchar_t>`

The virtual table for the `std::__timepunct<wchar_t>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::__timepunct<wchar_t>` class is described by [Table 12-148](#)

Table 12-148 typeid for `__timepunct<wchar_t>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>__timepunct<wchar_t></code>

12.1.91.2 Interfaces for Class `__timepunct<wchar_t>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::__timepunct<wchar_t>` specified in [Table 12-149](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-149 libstdcxx - Class `__timepunct<wchar_t>` Function Interfaces

<code>__timepunct<wchar_t>::_M_put(wchar_t*, unsigned long, wchar_t const*, tm const*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>__timepunct<wchar_t>::__timepunct(__locale_struct*, char const*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__timepunct<wchar_t>::__timepunct(__timepunct_cache<wchar_t>*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__timepunct<wchar_t>::__timepunct(unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__timepunct<wchar_t>::__timepunct(__locale_struct*, char const*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__timepunct<wchar_t>::__timepunct(__timepunct_cache<wchar_t>*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__timepunct<wchar_t>::__timepunct(unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]

12.1.92 Class `messages_base`

12.1.92.1 Class data for `messages_base`

The Run Time Type Information for the `std::messages_base` class is described by [Table 12-150](#)

Table 12-150 typeid for `messages_base`

Base Vtable	vtable for <code>__cxxabiv1::__class_type_info</code>
Name	typeid name for <code>messages_base</code>

12.1.92.2 Interfaces for Class `messages_base`

No external methods are defined for libstdcxx - Class `std::messages_base` in this part of the specification. See also the generic specification.

12.1.93 Class `messages<char>`

12.1.93.1 Class data for `messages<char>`

The virtual table for the `std::messages<char>` class is described in the generic part of this specification.

12.1.93.2 Interfaces for Class `messages<char>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::messages<char>` specified in [Table 12-151](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-151 libstdc++ - Class messages<char> Function Interfaces

messages<char>::messages(__locale_struct*, char const*, unsigned long) (GLIBCXX_3.4) [ISOCXX]
messages<char>::messages(unsigned long)(GLIBCXX_3.4) [ISOCXX]
messages<char>::messages(__locale_struct*, char const*, unsigned long) (GLIBCXX_3.4) [ISOCXX]
messages<char>::messages(unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.94 Class messages<wchar_t>

12.1.94.1 Class data for messages<wchar_t>

The virtual table for the std::messages<wchar_t> class is described in the generic part of this specification.

12.1.94.2 Interfaces for Class messages<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::messages<wchar_t> specified in [Table 12-152](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-152 libstdc++ - Class messages<wchar_t> Function Interfaces

messages<wchar_t>::messages(__locale_struct*, char const*, unsigned long) (GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::messages(unsigned long)(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::messages(__locale_struct*, char const*, unsigned long) (GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::messages(unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.95 Class messages_byname<char>

12.1.95.1 Class data for messages_byname<char>

The virtual table for the std::messages_byname<char> class is described in the generic part of this specification.

The Run Time Type Information for the std::messages_byname<char> class is described by [Table 12-153](#)

Table 12-153 typeid for messages_byname<char>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for messages_byname<char>

12.1.95.2 Interfaces for Class messages_byname<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::messages_byname<char> specified in [Table 12-154](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-154 libstdc++ - Class messages_byname<char> Function Interfaces

messages_byname<char>::messages_byname(char const*, unsigned long) (GLIBCXX_3.4) [ISOCXX]
messages_byname<char>::messages_byname(char const*, unsigned long) (GLIBCXX_3.4) [ISOCXX]

12.1.96 Class messages_byname<wchar_t>

12.1.96.1 Class data for messages_byname<wchar_t>

The virtual table for the std::messages_byname<wchar_t> class is described in the generic part of this specification.

The Run Time Type Information for the std::messages_byname<wchar_t> class is described by [Table 12-155](#)

Table 12-155 typeid for messages_byname<wchar_t>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for messages_byname<wchar_t>

12.1.96.2 Interfaces for Class messages_byname<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::messages_byname<wchar_t> specified in [Table 12-156](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-156 libstdc++ - Class messages_byname<wchar_t> Function Interfaces

messages_byname<wchar_t>::messages_byname(char const*, unsigned long) (GLIBCXX_3.4) [ISO CXX]
messages_byname<wchar_t>::messages_byname(char const*, unsigned long) (GLIBCXX_3.4) [ISO CXX]

12.1.97 Class numpunct<char>

12.1.97.1 Class data for numpunct<char>

The virtual table for the std::numpunct<char> class is described in the generic part of this specification.

The Run Time Type Information for the std::numpunct<char> class is described by [Table 12-157](#)

Table 12-157 typeid for numpunct<char>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for numpunct<char>

12.1.97.2 Interfaces for Class numpunct<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::numpunct<char> specified in [Table 12-158](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-158 libstdc++ - Class numpunct<char> Function Interfaces

numpunct<char>::numpunct(__locale_struct*, unsigned long)(GLIBCXX_3.4) [ISO CXX]
numpunct<char>::numpunct(__numpunct_cache<char>*, unsigned long) (GLIBCXX_3.4) [ISO CXX]
numpunct<char>::numpunct(unsigned long)(GLIBCXX_3.4) [ISO CXX]
numpunct<char>::numpunct(__locale_struct*, unsigned long)(GLIBCXX_3.4) [ISO CXX]

<code>numpunct<char>::numpunct(__numpunct_cache<char>*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>numpunct<char>::numpunct(unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]

12.1.98 Class `numpunct<wchar_t>`

12.1.98.1 Class data for `numpunct<wchar_t>`

The virtual table for the `std::numpunct<wchar_t>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::numpunct<wchar_t>` class is described by [Table 12-159](#)

Table 12-159 typeinfo for `numpunct<wchar_t>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>numpunct<wchar_t></code>

12.1.98.2 Interfaces for Class `numpunct<wchar_t>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::numpunct<wchar_t>` specified in [Table 12-160](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-160 libstdcxx - Class `numpunct<wchar_t>` Function Interfaces

<code>numpunct<wchar_t>::numpunct(__locale_struct*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>numpunct<wchar_t>::numpunct(unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>numpunct<wchar_t>::numpunct(__locale_struct*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>numpunct<wchar_t>::numpunct(unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]

12.1.99 Class `numpunct_byname<char>`

12.1.99.1 Class data for `numpunct_byname<char>`

The virtual table for the `std::numpunct_byname<char>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::numpunct_byname<char>` class is described by [Table 12-161](#)

Table 12-161 typeinfo for `numpunct_byname<char>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>numpunct_byname<char></code>

12.1.99.2 Interfaces for Class `numpunct_byname<char>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::numpunct_byname<char>` specified in [Table 12-162](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-162 libstdcxx - Class `numpunct_byname<char>` Function Interfaces

<code>numpunct_byname<char>::numpunct_byname(char const*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]

numpunct_byname<char>::numpunct_byname(char const*, unsigned long)
(GLIBCXX_3.4) [\[ISOCXX\]](#)

12.1.100 Class numpunct_byname<wchar_t>

12.1.100.1 Class data for numpunct_byname<wchar_t>

The virtual table for the std::numpunct_byname<wchar_t> class is described in the generic part of this specification.

The Run Time Type Information for the std::numpunct_byname<wchar_t> class is described by [Table 12-163](#)

Table 12-163 typeid for numpunct_byname<wchar_t>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for numpunct_byname<wchar_t>

12.1.100.2 Interfaces for Class numpunct_byname<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::numpunct_byname<wchar_t> specified in [Table 12-164](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-164 libstdcxx - Class numpunct_byname<wchar_t> Function Interfaces

numpunct_byname<wchar_t>::numpunct_byname(char const*, unsigned long)
(GLIBCXX_3.4) [\[ISOCXX\]](#)

numpunct_byname<wchar_t>::numpunct_byname(char const*, unsigned long)
(GLIBCXX_3.4) [\[ISOCXX\]](#)

12.1.101 Class __codecvt_abstract_base<char, char, __mbstate_t>

12.1.101.1 Class data for __codecvt_abstract_base<char, char, __mbstate_t>

The virtual table for the std::__codecvt_abstract_base<char, char, __mbstate_t> class is described in the generic part of this specification.

12.1.101.2 Interfaces for Class __codecvt_abstract_base<char, char, __mbstate_t>

No external methods are defined for libstdcxx - Class std::__codecvt_abstract_base<char, char, __mbstate_t> in this part of the specification. See also the generic specification.

12.1.102 Class __codecvt_abstract_base<wchar_t, char, __mbstate_t>

12.1.102.1 Class data for __codecvt_abstract_base<wchar_t, char, __mbstate_t>

The virtual table for the std::__codecvt_abstract_base<wchar_t, char, __mbstate_t> class is described in the generic part of this specification.

12.1.102.2 Interfaces for Class `__codecvt_abstract_base<wchar_t, char, __mbstate_t>`

No external methods are defined for `libstdcxx` - Class `std::__codecvt_abstract_base<wchar_t, char, __mbstate_t>` in this part of the specification. See also the generic specification.

12.1.103 Class `codecvt_base`

12.1.103.1 Class data for `codecvt_base`

The Run Time Type Information for the `std::codecvt_base` class is described by [Table 12-165](#)

Table 12-165 `typeinfo` for `codecvt_base`

Base Vtable	vtable for <code>__cxxabiv1::__class_type_info</code>
Name	<code>typeinfo</code> name for <code>codecvt_base</code>

12.1.103.2 Interfaces for Class `codecvt_base`

No external methods are defined for `libstdcxx` - Class `std::codecvt_base` in this part of the specification. See also the generic specification.

12.1.104 Class `codecvt<char, char, __mbstate_t>`

12.1.104.1 Class data for `codecvt<char, char, __mbstate_t>`

The virtual table for the `std::codecvt<char, char, __mbstate_t>` class is described by [Table 12-166](#)

Table 12-166 Primary vtable for `codecvt<char, char, __mbstate_t>`

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeinfo</code> for <code>codecvt<char, char, __mbstate_t></code>
<code>vfunc[0]:</code>	<code>codecvt<char, char, __mbstate_t>::~~codecvt()</code>
<code>vfunc[1]:</code>	<code>codecvt<char, char, __mbstate_t>::~~codecvt()</code>
<code>vfunc[2]:</code>	<code>codecvt<char, char, __mbstate_t>::do_out(__mbstate_t&, char const*, char const*, char const*&, char*, char*, char*&) const</code>
<code>vfunc[3]:</code>	<code>codecvt<char, char, __mbstate_t>::do_unshift(__mbstate_t&, char*, char*, char*&) const</code>
<code>vfunc[4]:</code>	<code>codecvt<char, char, __mbstate_t>::do_in(__mbstate_t&, char const*, char const*, char const*&, char*, char*, char*&) const</code>
<code>vfunc[5]:</code>	<code>codecvt<char, char, __mbstate_t>::do_encoding() const</code>
<code>vfunc[6]:</code>	<code>codecvt<char, char, __mbstate_t>::do_always_noconv() const</code>

vfunc[7]:	codecvt<char, char, __mbstate_t>::do_length(__mbstate_t&, char const*, char const*, unsigned long) const
vfunc[8]:	codecvt<char, char, __mbstate_t>::do_max_length() const

The Run Time Type Information for the std::codecvt<char, char, __mbstate_t> class is described by [Table 12-167](#)

Table 12-167 typeid for codecvt<char, char, __mbstate_t>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for codecvt<char, char, __mbstate_t>

12.1.104.2 Interfaces for Class codecvt<char, char, __mbstate_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::codecvt<char, char, __mbstate_t> specified in [Table 12-168](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-168 libstdc++ - Class codecvt<char, char, __mbstate_t> Function Interfaces

codecvt<char, char, __mbstate_t>::do_length(__mbstate_t&, char const*, char const*, unsigned long) const (GLIBCXX_3.4) [ISOCXX]
codecvt<char, char, __mbstate_t>::codecvt(__locale_struct*, unsigned long) (GLIBCXX_3.4) [ISOCXX]
codecvt<char, char, __mbstate_t>::codecvt(unsigned long) (GLIBCXX_3.4) [ISOCXX]
codecvt<char, char, __mbstate_t>::codecvt(__locale_struct*, unsigned long) (GLIBCXX_3.4) [ISOCXX]
codecvt<char, char, __mbstate_t>::codecvt(unsigned long) (GLIBCXX_3.4) [ISOCXX]

12.1.105 Class codecvt<wchar_t, char, __mbstate_t>

12.1.105.1 Class data for codecvt<wchar_t, char, __mbstate_t>

The virtual table for the std::codecvt<wchar_t, char, __mbstate_t> class is described by [Table 12-169](#)

Table 12-169 Primary vtable for codecvt<wchar_t, char, __mbstate_t>

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for codecvt<wchar_t, char, __mbstate_t>
vfunc[0]:	codecvt<wchar_t, char, __mbstate_t>::~~codecvt()
vfunc[1]:	codecvt<wchar_t, char, __mbstate_t>::~~codecvt()
vfunc[2]:	codecvt<wchar_t, char, __mbstate_t>::do_out(__mbstate_t&, wchar_t const*, wchar_t const*, wchar_t

	const*&, char*, char*, char*&) const
vfunc[3]:	codecvt<wchar_t, char, __mbstate_t>::do_unshift(__mbstate_t&, char*, char*, char*&) const
vfunc[4]:	codecvt<wchar_t, char, __mbstate_t>::do_in(__mbstate_t&, char const*, char const*, char const*&, wchar_t*, wchar_t*, wchar_t*&) const
vfunc[5]:	codecvt<wchar_t, char, __mbstate_t>::do_encoding() const
vfunc[6]:	codecvt<wchar_t, char, __mbstate_t>::do_always_noconv() const
vfunc[7]:	codecvt<wchar_t, char, __mbstate_t>::do_length(__mbstate_t&, char const*, char const*, unsigned long) const
vfunc[8]:	codecvt<wchar_t, char, __mbstate_t>::do_max_length() const

The Run Time Type Information for the `std::codecvt<wchar_t, char, __mbstate_t>` class is described by [Table 12-170](#)

Table 12-170 typeid for `codecvt<wchar_t, char, __mbstate_t>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>codecvt<wchar_t, char, __mbstate_t></code>

12.1.105.2 Interfaces for Class `codecvt<wchar_t, char, __mbstate_t>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::codecvt<wchar_t, char, __mbstate_t>` specified in [Table 12-171](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-171 `libstdc++` - Class `codecvt<wchar_t, char, __mbstate_t>` Function Interfaces

<code>codecvt<wchar_t, char, __mbstate_t>::do_length(__mbstate_t&, char const*, char const*, unsigned long) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>codecvt<wchar_t, char, __mbstate_t>::codecvt(__locale_struct*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>codecvt<wchar_t, char, __mbstate_t>::codecvt(unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>codecvt<wchar_t, char, __mbstate_t>::codecvt(__locale_struct*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>codecvt<wchar_t, char, __mbstate_t>::codecvt(unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]

12.1.106 Class `codecvt_byname<char, char, __mbstate_t>`

12.1.106.1 Class data for `codecvt_byname<char, char, __mbstate_t>`

The virtual table for the `std::codecvt_byname<char, char, __mbstate_t>` class is described by [Table 12-172](#)

Table 12-172 Primary vtable for `codecvt_byname<char, char, __mbstate_t>`

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>codecvt_byname<char, char, __mbstate_t></code>
vfunc[0]:	<code>codecvt_byname<char, char, __mbstate_t>::~~codecvt_byname()</code>
vfunc[1]:	<code>codecvt_byname<char, char, __mbstate_t>::~~codecvt_byname()</code>
vfunc[2]:	<code>codecvt<char, char, __mbstate_t>::do_out(__mbstate_t&, char const*, char const*, char const*&, char*, char*, char*&) const</code>
vfunc[3]:	<code>codecvt<char, char, __mbstate_t>::do_unshift(__mbstate_t&, char*, char*, char*&) const</code>
vfunc[4]:	<code>codecvt<char, char, __mbstate_t>::do_in(__mbstate_t&, char const*, char const*, char const*&, char*, char*, char*&) const</code>
vfunc[5]:	<code>codecvt<char, char, __mbstate_t>::do_encoding() const</code>
vfunc[6]:	<code>codecvt<char, char, __mbstate_t>::do_always_noconv() const</code>
vfunc[7]:	<code>codecvt<char, char, __mbstate_t>::do_length(__mbstate_t&, char const*, char const*, unsigned long) const</code>
vfunc[8]:	<code>codecvt<char, char, __mbstate_t>::do_max_length() const</code>

The Run Time Type Information for the `std::codecvt_byname<char, char, __mbstate_t>` class is described by [Table 12-173](#)

Table 12-173 typeinfo for `codecvt_byname<char, char, __mbstate_t>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>codecvt_byname<char, char, __mbstate_t></code>

12.1.106.2 Interfaces for Class `codecvt_byname<char, char, __mbstate_t>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::codecvt_byname<char, char, __mbstate_t>` specified in [Table 12-174](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-174 libstdcxx - Class `codecvt_byname<char, char, __mbstate_t>` Function Interfaces

<code>codecvt_byname<char, char, __mbstate_t>::codecvt_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt_byname<char, char, __mbstate_t>::codecvt_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.107 Class `codecvt_byname<wchar_t, char, __mbstate_t>`

12.1.107.1 Class data for `codecvt_byname<wchar_t, char, __mbstate_t>`

The virtual table for the `std::codecvt_byname<wchar_t, char, __mbstate_t>` class is described by [Table 12-175](#)

Table 12-175 Primary vtable for `codecvt_byname<wchar_t, char, __mbstate_t>`

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeinfo for codecvt_byname<wchar_t, char, __mbstate_t></code>
<code>vfunc[0]:</code>	<code>codecvt_byname<wchar_t, char, __mbstate_t>::~~codecvt_byname()</code>
<code>vfunc[1]:</code>	<code>codecvt_byname<wchar_t, char, __mbstate_t>::~~codecvt_byname()</code>
<code>vfunc[2]:</code>	<code>codecvt<wchar_t, char, __mbstate_t>::do_out(__mbstate_t&, wchar_t const*, wchar_t const*, wchar_t const*&, char*, char*, char*&) const</code>
<code>vfunc[3]:</code>	<code>codecvt<wchar_t, char, __mbstate_t>::do_unshift(__mbstate_t&, char*, char*, char*&) const</code>
<code>vfunc[4]:</code>	<code>codecvt<wchar_t, char, __mbstate_t>::do_in(__mbstate_t&, char const*, char const*, char const*&, wchar_t*, wchar_t*, wchar_t*&) const</code>
<code>vfunc[5]:</code>	<code>codecvt<wchar_t, char, __mbstate_t>::do_encoding() const</code>
<code>vfunc[6]:</code>	<code>codecvt<wchar_t, char, __mbstate_t>::do_always_noconv() const</code>
<code>vfunc[7]:</code>	<code>codecvt<wchar_t, char, __mbstate_t>::do_length(__mbstate_t&, char const*, char const*, unsigned long) const</code>
<code>vfunc[8]:</code>	<code>codecvt<wchar_t, char, __mbstate_t>::do_max_length() const</code>

The Run Time Type Information for the `std::codecvt_byname<wchar_t, char, __mbstate_t>` class is described by [Table 12-176](#)

Table 12-176 typeinfo for `codecvt_byname<wchar_t, char, __mbstate_t>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>codecvt_byname<wchar_t, char, __mbstate_t></code>

12.1.107.2 Interfaces for Class `codecvt_byname<wchar_t, char, __mbstate_t>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::codecvt_byname<wchar_t, char, __mbstate_t>` specified in [Table 12-177](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-177 libstdcxx - Class `codecvt_byname<wchar_t, char, __mbstate_t>` Function Interfaces

<code>codecvt_byname<wchar_t, char, __mbstate_t>::codecvt_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt_byname<wchar_t, char, __mbstate_t>::codecvt_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.108 Class `collate<char>`

12.1.108.1 Class data for `collate<char>`

The virtual table for the `std::collate<char>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::collate<char>` class is described by [Table 12-178](#)

Table 12-178 typeinfo for `collate<char>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>collate<char></code>

12.1.108.2 Interfaces for Class `collate<char>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::collate<char>` specified in [Table 12-179](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-179 libstdcxx - Class `collate<char>` Function Interfaces

<code>collate<char>::_M_transform(char*, char const*, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate<char>::collate(__locale_struct*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate<char>::collate(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate<char>::collate(__locale_struct*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate<char>::collate(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.109 Class `collate<wchar_t>`

12.1.109.1 Class data for `collate<wchar_t>`

The virtual table for the `std::collate<wchar_t>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::collate<wchar_t>` class is described by [Table 12-180](#)

Table 12-180 `typeinfo` for `collate<wchar_t>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	<code>typeinfo</code> name for <code>collate<wchar_t></code>

12.1.109.2 Interfaces for Class `collate<wchar_t>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::collate<wchar_t>` specified in [Table 12-181](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-181 `libstdcxx` - Class `collate<wchar_t>` Function Interfaces

<code>collate<wchar_t>::_M_transform(wchar_t*, wchar_t const*, unsigned long)</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate<wchar_t>::collate(__locale_struct*, unsigned long)(GLIBCXX_3.4)</code> <code>[ISOCXX]</code>
<code>collate<wchar_t>::collate(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate<wchar_t>::collate(__locale_struct*, unsigned long)(GLIBCXX_3.4)</code> <code>[ISOCXX]</code>
<code>collate<wchar_t>::collate(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.110 Class `collate_byname<char>`

12.1.110.1 Class data for `collate_byname<char>`

The virtual table for the `std::collate_byname<char>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::collate_byname<char>` class is described by [Table 12-182](#)

Table 12-182 `typeinfo` for `collate_byname<char>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	<code>typeinfo</code> name for <code>collate_byname<char></code>

12.1.110.2 Interfaces for Class `collate_byname<char>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::collate_byname<char>` specified in [Table 12-183](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-183 `libstdcxx` - Class `collate_byname<char>` Function Interfaces

<code>collate_byname<char>::collate_byname(char const*, unsigned long)</code> <code>(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate_byname<char>::collate_byname(char const*, unsigned long)</code>

(GLIBCXX_3.4) [\[ISOCXX\]](#)

12.1.111 Class `collate_byname<wchar_t>`

12.1.111.1 Class data for `collate_byname<wchar_t>`

The virtual table for the `std::collate_byname<wchar_t>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::collate_byname<wchar_t>` class is described by [Table 12-184](#)

Table 12-184 typeinfo for `collate_byname<wchar_t>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>collate_byname<wchar_t></code>

12.1.111.2 Interfaces for Class `collate_byname<wchar_t>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::collate_byname<wchar_t>` specified in [Table 12-185](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-185 libstdcxx - Class `collate_byname<wchar_t>` Function Interfaces

<code>collate_byname<wchar_t>::collate_byname(char const*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]
<code>collate_byname<wchar_t>::collate_byname(char const*, unsigned long)</code> (GLIBCXX_3.4) [ISOCXX]

12.1.112 Class `time_base`

12.1.112.1 Class data for `time_base`

The Run Time Type Information for the `std::time_base` class is described by [Table 12-186](#)

Table 12-186 typeinfo for `time_base`

Base Vtable	vtable for <code>__cxxabiv1::__class_type_info</code>
Name	typeinfo name for <code>time_base</code>

12.1.112.2 Interfaces for Class `time_base`

No external methods are defined for libstdcxx - Class `std::time_base` in this part of the specification. See also the generic specification.

12.1.113 Class `time_get_byname<char, istreambuf_iterator<char, char_traits<char>>>`

12.1.113.1 Class data for `time_get_byname<char, istreambuf_iterator<char, char_traits<char>>>`

The virtual table for the `std::time_get_byname<char, std::istreambuf_iterator<char, std::char_traits<char>>>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::time_get_byname<char, std::istreambuf_iterator<char, std::char_traits<char>>>` class is described by [Table 12-187](#)

Table 12-187 typeid for `time_get_byname<char, istreambuf_iterator<char, char_traits<char>>>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>time_get_byname<char, istreambuf_iterator<char, char_traits<char>>></code>

12.1.113.2 Interfaces for Class `time_get_byname<char, istreambuf_iterator<char, char_traits<char>>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::time_get_byname<char, std::istreambuf_iterator<char, std::char_traits<char>>>` specified in [Table 12-188](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-188 `libstdc++` - Class `time_get_byname<char, istreambuf_iterator<char, char_traits<char>>>` Function Interfaces

<code>time_get_byname<char, istreambuf_iterator<char, char_traits<char>>></code> <code>>::time_get_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get_byname<char, istreambuf_iterator<char, char_traits<char>>></code> <code>>::time_get_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.114 Class `time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`**12.1.114.1 Class data for `time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`**

The virtual table for the `std::time_get_byname<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::time_get_byname<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` class is described by [Table 12-189](#)

Table 12-189 typeid for `time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>></code>

12.1.114.2 Interfaces for Class `time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::time_get_byname<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` specified in [Table 12-190](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-190 libstdcxx - Class time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>> Function Interfaces

time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>::time_get_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>::time_get_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.115 Class time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>>

12.1.115.1 Class data for time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>>

The virtual table for the std::time_put_byname<char, std::ostreambuf_iterator<char, std::char_traits<char>>>> class is described in the generic part of this specification.

The Run Time Type Information for the std::time_put_byname<char, std::ostreambuf_iterator<char, std::char_traits<char>>>> class is described by [Table 12-191](#)

Table 12-191 typeid for time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>>

12.1.115.2 Interfaces for Class time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::time_put_byname<char, std::ostreambuf_iterator<char, std::char_traits<char>>>> specified in [Table 12-192](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-192 libstdcxx - Class time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>> Function Interfaces

time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>>::time_put_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>>::time_put_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.116 Class time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>>

12.1.116.1 Class data for time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>>

The virtual table for the std::time_put_byname<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>> class is described in the generic part of this specification.

The Run Time Type Information for the std::time_put_byname<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>> class is described by [Table 12-193](#)

Table 12-193 typeinfo for time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>

12.1.116.2 Interfaces for Class time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>

An LSB conforming implementation shall provide the architecture specific methods for Class `std::time_put_byname<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` specified in [Table 12-194](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-194 libstdcxx - Class time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>> Function Interfaces

<code>time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>::time_put_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>::time_put_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.117 Class time_get<char, istreambuf_iterator<char, char_traits<char>>>

12.1.117.1 Class data for time_get<char, istreambuf_iterator<char, char_traits<char>>>

The virtual table for the `std::time_get<char, std::istreambuf_iterator<char, std::char_traits<char>>>` class is described in the generic part of this specification.

12.1.117.2 Interfaces for Class time_get<char, istreambuf_iterator<char, char_traits<char>>>

An LSB conforming implementation shall provide the architecture specific methods for Class `std::time_get<char, std::istreambuf_iterator<char, std::char_traits<char>>>` specified in [Table 12-195](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-195 libstdcxx - Class time_get<char, istreambuf_iterator<char, char_traits<char>>> Function Interfaces

<code>time_get<char, istreambuf_iterator<char, char_traits<char>>>::_M_extract_num(istreambuf_iterator<char, char_traits<char>>, istreambuf_iterator<char, char_traits<char>>, int&, int, int, unsigned long, ios_base&, _Ios_Iostate&) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get<char, istreambuf_iterator<char, char_traits<char>>>::_M_extract_name(istreambuf_iterator<char, char_traits<char>>, istreambuf_iterator<char, char_traits<char>>, int&, char const**, unsigned long, ios_base&, _Ios_Iostate&) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get<char, istreambuf_iterator<char, char_traits<char>>>::time_get(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get<char, istreambuf_iterator<char, char_traits<char>>>::time_get(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.118 Class `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

12.1.118.1 Class data for `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

The virtual table for the `std::time_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` class is described in the generic part of this specification.

12.1.118.2 Interfaces for Class `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::time_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` specified in [Table 12-196](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-196 libstdcxx - Class `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>` Function Interfaces

<code>time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>::M_extract_num(istreambuf_iterator<wchar_t, char_traits<wchar_t>>, istreambuf_iterator<wchar_t, char_traits<wchar_t>>, int&, int, int, unsigned long, ios_base&, _Ios_Iostate&) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>::M_extract_name(istreambuf_iterator<wchar_t, char_traits<wchar_t>>, istreambuf_iterator<wchar_t, char_traits<wchar_t>>, int&, wchar_t const**, unsigned long, ios_base&, _Ios_Iostate&) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>::time_get(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>::time_get(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.119 Class `time_put<char, ostreambuf_iterator<char, char_traits<char>>>`

12.1.119.1 Class data for `time_put<char, ostreambuf_iterator<char, char_traits<char>>>`

The virtual table for the `std::time_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::time_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>>` class is described by [Table 12-197](#)

Table 12-197 typeid for `time_put<char, ostreambuf_iterator<char, char_traits<char>>>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>time_put<char, ostreambuf_iterator<char, char_traits<char>>></code>
flags:	8

basetype:	typeinfo for locale::facet	2
basetype:	typeinfo for time_base	2

12.1.119.2 Interfaces for Class time_put<char, ostreambuf_iterator<char, char_traits<char>>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::time_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>> specified in [Table 12-198](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-198 libstdcxx - Class time_put<char, ostreambuf_iterator<char, char_traits<char>>> Function Interfaces

time_put<char, ostreambuf_iterator<char, char_traits<char>>>::time_put(unsigned long)(GLIBCXX_3.4) [ISOCXX1]
time_put<char, ostreambuf_iterator<char, char_traits<char>>>::time_put(unsigned long)(GLIBCXX_3.4) [ISOCXX1]

12.1.120 Class time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>

12.1.120.1 Class data for time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>

The virtual table for the std::time_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>> class is described in the generic part of this specification.

The Run Time Type Information for the std::time_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>> class is described by [Table 12-199](#)

Table 12-199 typeinfo for time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info	
Name	typeinfo name for time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>	
flags:	8	
basetype:	typeinfo for locale::facet	2
basetype:	typeinfo for time_base	2

12.1.120.2 Interfaces for Class time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::time_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>> specified in [Table 12-200](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-200 libstdcxx - Class `time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>` Function Interfaces

<code>time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>::time_put(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>::time_put(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.121 Class `money_punct<char, false>`

12.1.121.1 Class data for `money_punct<char, false>`

The virtual table for the `std::money_punct<char, false>` class is described in the generic part of this specification.

12.1.121.2 Interfaces for Class `money_punct<char, false>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::money_punct<char, false>` specified in [Table 12-201](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-201 libstdcxx - Class `money_punct<char, false>` Function Interfaces

<code>money_punct<char, false>::money_punct(__locale_struct*, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct<char, false>::money_punct(__money_punct_cache<char, false>*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct<char, false>::money_punct(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct<char, false>::money_punct(__locale_struct*, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct<char, false>::money_punct(__money_punct_cache<char, false>*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct<char, false>::money_punct(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.122 Class `money_punct<char, true>`

12.1.122.1 Class data for `money_punct<char, true>`

The virtual table for the `std::money_punct<char, true>` class is described in the generic part of this specification.

12.1.122.2 Interfaces for Class `money_punct<char, true>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::money_punct<char, true>` specified in [Table 12-202](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-202 libstdcxx - Class `money_punct<char, true>` Function Interfaces

<code>money_punct<char, true>::money_punct(__locale_struct*, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct<char, true>::money_punct(__money_punct_cache<char, true>*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct<char, true>::money_punct(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct<char, true>::money_punct(__locale_struct*, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct<char, true>::money_punct(__money_punct_cache<char, true>*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

money_punct<char, true>::money_punct(unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.123 Class money_punct<wchar_t, false>

12.1.123.1 Class data for money_punct<wchar_t, false>

The virtual table for the std::money_punct<wchar_t, false> class is described in the generic part of this specification.

12.1.123.2 Interfaces for Class money_punct<wchar_t, false>

An LSB conforming implementation shall provide the architecture specific methods for Class std::money_punct<wchar_t, false> specified in [Table 12-203](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-203 libstdcxx - Class money_punct<wchar_t, false> Function Interfaces

money_punct<wchar_t, false>::money_punct(__locale_struct*, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::money_punct(__money_punct_cache<wchar_t, false>*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::money_punct(unsigned long)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::money_punct(__locale_struct*, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::money_punct(__money_punct_cache<wchar_t, false>*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::money_punct(unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.124 Class money_punct<wchar_t, true>

12.1.124.1 Class data for money_punct<wchar_t, true>

The virtual table for the std::money_punct<wchar_t, true> class is described in the generic part of this specification.

12.1.124.2 Interfaces for Class money_punct<wchar_t, true>

An LSB conforming implementation shall provide the architecture specific methods for Class std::money_punct<wchar_t, true> specified in [Table 12-204](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-204 libstdcxx - Class money_punct<wchar_t, true> Function Interfaces

money_punct<wchar_t, true>::money_punct(__locale_struct*, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::money_punct(__money_punct_cache<wchar_t, true>*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::money_punct(unsigned long)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::money_punct(__locale_struct*, char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::money_punct(__money_punct_cache<wchar_t, true>*, unsigned long)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::money_punct(unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.125 Class `money_punct_byname<char, false>`

12.1.125.1 Class data for `money_punct_byname<char, false>`

The virtual table for the `std::money_punct_byname<char, false>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::money_punct_byname<char, false>` class is described by [Table 12-205](#)

Table 12-205 `typeinfo` for `money_punct_byname<char, false>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>money_punct_byname<char, false></code>

12.1.125.2 Interfaces for Class `money_punct_byname<char, false>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::money_punct_byname<char, false>` specified in [Table 12-206](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-206 `libstdcxx` - Class `money_punct_byname<char, false>` Function Interfaces

<code>money_punct_byname<char, false>::money_punct_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct_byname<char, false>::money_punct_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.126 Class `money_punct_byname<char, true>`

12.1.126.1 Class data for `money_punct_byname<char, true>`

The virtual table for the `std::money_punct_byname<char, true>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::money_punct_byname<char, true>` class is described by [Table 12-207](#)

Table 12-207 `typeinfo` for `money_punct_byname<char, true>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>money_punct_byname<char, true></code>

12.1.126.2 Interfaces for Class `money_punct_byname<char, true>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::money_punct_byname<char, true>` specified in [Table 12-208](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-208 `libstdcxx` - Class `money_punct_byname<char, true>` Function Interfaces

<code>money_punct_byname<char, true>::money_punct_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct_byname<char, true>::money_punct_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.127 Class `moneypunct_byname<wchar_t, false>`

12.1.127.1 Class data for `moneypunct_byname<wchar_t, false>`

The virtual table for the `std::moneypunct_byname<wchar_t, false>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::moneypunct_byname<wchar_t, false>` class is described by [Table 12-209](#)

Table 12-209 `typeinfo` for `moneypunct_byname<wchar_t, false>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>moneypunct_byname<wchar_t, false></code>

12.1.127.2 Interfaces for Class `moneypunct_byname<wchar_t, false>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::moneypunct_byname<wchar_t, false>` specified in [Table 12-210](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-210 `libstdcxx` - Class `moneypunct_byname<wchar_t, false>` Function Interfaces

<code>moneypunct_byname<wchar_t, false>::moneypunct_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>moneypunct_byname<wchar_t, false>::moneypunct_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.128 Class `moneypunct_byname<wchar_t, true>`

12.1.128.1 Class data for `moneypunct_byname<wchar_t, true>`

The virtual table for the `std::moneypunct_byname<wchar_t, true>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::moneypunct_byname<wchar_t, true>` class is described by [Table 12-211](#)

Table 12-211 `typeinfo` for `moneypunct_byname<wchar_t, true>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>moneypunct_byname<wchar_t, true></code>

12.1.128.2 Interfaces for Class `moneypunct_byname<wchar_t, true>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::moneypunct_byname<wchar_t, true>` specified in [Table 12-212](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-212 `libstdcxx` - Class `moneypunct_byname<wchar_t, true>` Function Interfaces

<code>moneypunct_byname<wchar_t, true>::moneypunct_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>moneypunct_byname<wchar_t, true>::moneypunct_byname(char const*, unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.129 Class money_base

12.1.129.1 Class data for money_base

The Run Time Type Information for the `std::money_base` class is described by [Table 12-213](#)

Table 12-213 typeid for money_base

Base Vtable	vtable for <code>__cxxabiv1::__class_type_info</code>
Name	typeid name for <code>money_base</code>

12.1.129.2 Interfaces for Class money_base

No external methods are defined for `libstdcxx` - Class `std::money_base` in this part of the specification. See also the generic specification.

12.1.130 Class money_get<char, istreambuf_iterator<char, char_traits<char>>>

12.1.130.1 Class data for money_get<char, istreambuf_iterator<char, char_traits<char>>>

The virtual table for the `std::money_get<char, std::istreambuf_iterator<char, std::char_traits<char>>>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::money_get<char, std::istreambuf_iterator<char, std::char_traits<char>>>` class is described by [Table 12-214](#)

Table 12-214 typeid for money_get<char, istreambuf_iterator<char, char_traits<char>>>

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>money_get<char, istreambuf_iterator<char, char_traits<char>>></code>

12.1.130.2 Interfaces for Class money_get<char, istreambuf_iterator<char, char_traits<char>>>

An LSB conforming implementation shall provide the architecture specific methods for Class `std::money_get<char, std::istreambuf_iterator<char, std::char_traits<char>>>` specified in [Table 12-215](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-215 libstdcxx - Class money_get<char, istreambuf_iterator<char, char_traits<char>>> Function Interfaces

<code>money_get<char, istreambuf_iterator<char, char_traits<char>>></code> <code>>::money_get(unsigned long)(GLIBCXX_3.4) [ISOXXX]</code>
<code>money_get<char, istreambuf_iterator<char, char_traits<char>>></code> <code>>::money_get(unsigned long)(GLIBCXX_3.4) [ISOXXX]</code>

12.1.131 Class `money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

12.1.131.1 Class data for `money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

The virtual table for the `std::money_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::money_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` class is described by [Table 12-216](#)

Table 12-216 `typeinfo` for `money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	<code>typeinfo</code> name for <code>money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>></code>

12.1.131.2 Interfaces for Class `money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::money_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` specified in [Table 12-217](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-217 `libstdcxx` - Class `money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>` Function Interfaces

<code>money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>::money_get(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>::money_get(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.132 Class `money_put<char, ostreambuf_iterator<char, char_traits<char>>>`

12.1.132.1 Class data for `money_put<char, ostreambuf_iterator<char, char_traits<char>>>`

The virtual table for the `std::money_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::money_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>>` class is described by [Table 12-218](#)

Table 12-218 `typeinfo` for `money_put<char, ostreambuf_iterator<char, char_traits<char>>>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	<code>typeinfo</code> name for <code>money_put<char, ostreambuf_iterator<char, char_traits<char>>></code>

12.1.132.2 Interfaces for Class `money_put<char, ostreambuf_iterator<char, char_traits<char>>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::money_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>>` specified in [Table 12-219](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-219 libstdcxx - Class `money_put<char, ostreambuf_iterator<char, char_traits<char>>>` Function Interfaces

<code>money_put<char, ostreambuf_iterator<char, char_traits<char>>></code> <code>>::money_put(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_put<char, ostreambuf_iterator<char, char_traits<char>>></code> <code>>::money_put(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.133 Class `money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

12.1.133.1 Class data for `money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

The virtual table for the `std::money_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` class is described in the generic part of this specification.

The Run Time Type Information for the `std::money_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` class is described by [Table 12-220](#)

Table 12-220 typeid for `money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>></code>

12.1.133.2 Interfaces for Class `money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::money_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` specified in [Table 12-221](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-221 libstdcxx - Class `money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>` Function Interfaces

<code>money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>></code> <code>>::money_put(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>></code> <code>>::money_put(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.134 Class locale

12.1.134.1 Interfaces for Class locale

An LSB conforming implementation shall provide the architecture specific methods for Class `std::locale` specified in [Table 12-222](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-222 libstdcxx - Class locale Function Interfaces

locale::_Impl::_Impl(char const*, unsigned long)(GLIBCXX_3.4) [LSB]
locale::_Impl::_Impl(locale::_Impl const&, unsigned long)(GLIBCXX_3.4) [LSB]
locale::_Impl::_Impl(unsigned long)(GLIBCXX_3.4) [LSB]
locale::_Impl::_Impl(char const*, unsigned long)(GLIBCXX_3.4) [LSB]
locale::_Impl::_Impl(locale::_Impl const&, unsigned long)(GLIBCXX_3.4) [LSB]
locale::_Impl::_Impl(unsigned long)(GLIBCXX_3.4) [LSB]

12.1.135 Class locale::facet

12.1.135.1 Class data for locale::facet

The virtual table for the std::locale::facet class is described in the generic part of this specification.

The Run Time Type Information for the std::locale::facet class is described by [Table 12-223](#)

Table 12-223 typeid for locale::facet

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeid name for locale::facet

12.1.135.2 Interfaces for Class locale::facet

No external methods are defined for libstdcxx - Class std::locale::facet in this part of the specification. See also the generic specification.

12.1.136 facet functions

12.1.136.1 Interfaces for facet functions

No external methods are defined for libstdcxx - facet functions in this part of the specification. See also the generic specification.

12.1.137 Class __num_base

12.1.137.1 Class data for __num_base

12.1.137.2 Interfaces for Class __num_base

No external methods are defined for libstdcxx - Class std::__num_base in this part of the specification. See also the generic specification.

12.1.138 Class num_get<char, istreambuf_iterator<char, char_traits<char>>>

12.1.138.1 Class data for num_get<char, istreambuf_iterator<char, char_traits<char>>>

The virtual table for the std::num_get<char, std::istreambuf_iterator<char, std::char_traits<char>>> class is described in the generic part of this specification.

The Run Time Type Information for the std::num_get<char, std::istreambuf_iterator<char, std::char_traits<char>>> class is described by [Table 12-224](#)

Table 12-224 typeid for num_get<char, istreambuf_iterator<char, char_traits<char>>>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for num_get<char, istreambuf_iterator<char, char_traits<char>>>
basetype:	typeid for locale::facet

12.1.138.2 Interfaces for Class num_get<char, istreambuf_iterator<char, char_traits<char>>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::num_get<char, std::istreambuf_iterator<char, std::char_traits<char>>> specified in [Table 12-225](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-225 libstdcxx - Class num_get<char, istreambuf_iterator<char, char_traits<char>>> Function Interfaces

num_get<char, istreambuf_iterator<char, char_traits<char>>>::num_get(unsigned long)(GLIBCXX_3.4) [ISO CXX]
num_get<char, istreambuf_iterator<char, char_traits<char>>>::num_get(unsigned long)(GLIBCXX_3.4) [ISO CXX]

12.1.139 Class num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>

12.1.139.1 Class data for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>

The virtual table for the std::num_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>> class is described in the generic part of this specification.

The Run Time Type Information for the std::num_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>> class is described by [Table 12-226](#)

Table 12-226 typeid for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>
basetype:	typeid for locale::facet

12.1.139.2 Interfaces for Class num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::num_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>> specified in [Table 12-227](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-227 libstdcxx - Class num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>> Function Interfaces

num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>> >::num_get(unsigned long)(GLIBCXX_3.4) [ISOCXX]
num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>> >::num_get(unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.140 Class num_put<char, ostreambuf_iterator<char, char_traits<char>>>

12.1.140.1 Class data for num_put<char, ostreambuf_iterator<char, char_traits<char>>>

The virtual table for the std::num_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>> class is described in the generic part of this specification.

The Run Time Type Information for the std::num_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>> class is described by [Table 12-228](#)

Table 12-228 typeid for num_put<char, ostreambuf_iterator<char, char_traits<char>>>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for num_put<char, ostreambuf_iterator<char, char_traits<char>>>
basetype:	typeid for locale::facet

12.1.140.2 Interfaces for Class num_put<char, ostreambuf_iterator<char, char_traits<char>>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::num_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>> specified in [Table 12-229](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-229 libstdcxx - Class num_put<char, ostreambuf_iterator<char, char_traits<char>>> Function Interfaces

num_put<char, ostreambuf_iterator<char, char_traits<char>>>::_M_group_int(char const*, unsigned long, char, ios_base&, char*, char*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<char, ostreambuf_iterator<char, char_traits<char>>> >::_M_group_float(char const*, unsigned long, char, char const*, char*, char*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<char, ostreambuf_iterator<char, char_traits<char>>>::_M_pad(char, long, ios_base&, char*, char const*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<char, ostreambuf_iterator<char, char_traits<char>>>::num_put(unsigned long)(GLIBCXX_3.4) [ISOCXX]
num_put<char, ostreambuf_iterator<char, char_traits<char>>>::num_put(unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.141 Class num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>

12.1.141.1 Class data for num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>

The virtual table for the std::num_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>> class is described in the generic part of this specification.

The Run Time Type Information for the std::num_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>> class is described by [Table 12-230](#)

Table 12-230 typeid for num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>
basetype:	typeid for locale::facet

12.1.141.2 Interfaces for Class num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>

An LSB conforming implementation shall provide the architecture specific methods for Class std::num_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t>>> specified in [Table 12-231](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-231 libstdc++ - Class num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>> Function Interfaces

num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>::_M_group_int(char const*, unsigned long, wchar_t, ios_base&, wchar_t*, wchar_t*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>::_M_group_float(char const*, unsigned long, wchar_t, wchar_t const*, wchar_t*, wchar_t*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>::_M_pad(wchar_t, long, ios_base&, wchar_t*, wchar_t const*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>::num_put(unsigned long)(GLIBCXX_3.4) [ISOCXX]
num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>::num_put(unsigned long)(GLIBCXX_3.4) [ISOCXX]

12.1.142 Class gslice

12.1.142.1 Class data for gslice

12.1.142.2 Interfaces for Class gslice

An LSB conforming implementation shall provide the architecture specific methods for Class std::gslice specified in [Table 12-232](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-232 libstdcxx - Class `gslice` Function Interfaces

<code>gslice::_Indexer::_Indexer(unsigned long, valarray<unsigned long> const&, valarray<unsigned long> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>gslice::_Indexer::_Indexer(unsigned long, valarray<unsigned long> const&, valarray<unsigned long> const&)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.143 Class `__basic_file<char>`

12.1.143.1 Class data for `__basic_file<char>`

12.1.143.2 Interfaces for Class `__basic_file<char>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::__basic_file<char>` specified in [Table 12-233](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-233 libstdcxx - Class `__basic_file<char>` Function Interfaces

<code>__basic_file<char>::xsgetn(char*, long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__basic_file<char>::xsputn(char const*, long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__basic_file<char>::seekoff(long, _Ios_Seekdir)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__basic_file<char>::xspun_2(char const*, long, char const*, long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.144 Class `_List_node_base`

12.1.144.1 Interfaces for Class `_List_node_base`

No external methods are defined for `libstdcxx - Class std::_List_node_base` in this part of the specification. See also the generic specification.

12.1.145 Class `valarray<unsigned int>`

12.1.145.1 Class data for `valarray<unsigned int>`

12.1.145.2 Interfaces for Class `valarray<unsigned int>`

An LSB conforming implementation shall provide the architecture specific methods for Class `std::valarray<unsigned int>` specified in [Table 12-234](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-234 libstdcxx - Class `valarray<unsigned int>` Function Interfaces

<code>valarray<unsigned long>::size() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>valarray<unsigned long>::valarray(valarray<unsigned long> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>valarray<unsigned long>::valarray(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>valarray<unsigned long>::valarray(valarray<unsigned long> const&)(GLIBCXX_3.4) [ISOCXX]</code>
<code>valarray<unsigned long>::valarray(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>valarray<unsigned long>::~valarray()(GLIBCXX_3.4) [ISOCXX]</code>
<code>valarray<unsigned long>::~valarray()(GLIBCXX_3.4) [ISOCXX]</code>
<code>valarray<unsigned long>::operator[](unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>

12.1.146 Class allocator<char>

12.1.146.1 Class data for allocator<char>

12.1.146.2 Interfaces for Class allocator<char>

No external methods are defined for libstdcxx - Class std::allocator<char> in this part of the specification. See also the generic specification.

12.1.147 Class allocator<wchar_t>

12.1.147.1 Class data for allocator<wchar_t>

12.1.147.2 Interfaces for Class allocator<wchar_t>

No external methods are defined for libstdcxx - Class std::allocator<wchar_t> in this part of the specification. See also the generic specification.

12.1.148 Class `__gnu_cxx::__pool<true>`

12.1.148.1 Interfaces for Class `__gnu_cxx::__pool<true>`

An LSB conforming implementation shall provide the architecture specific methods for Class `__gnu_cxx::__pool<true>` specified in [Table 12-235](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-235 libstdcxx - Class `__gnu_cxx::__pool<true>` Function Interfaces

<code>__gnu_cxx::__pool<true>::_M_reclaim_block(char*, unsigned long)</code> (GLIBCXX_3.4.4) [LSB]
<code>__gnu_cxx::__pool<true>::_M_reserve_block(unsigned long, unsigned long)</code> (GLIBCXX_3.4.4) [LSB]

12.1.149 Class `__gnu_cxx::__pool<false>`

12.1.149.1 Interfaces for Class `__gnu_cxx::__pool<false>`

An LSB conforming implementation shall provide the architecture specific methods for Class `__gnu_cxx::__pool<false>` specified in [Table 12-236](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-236 libstdcxx - Class `__gnu_cxx::__pool<false>` Function Interfaces

<code>__gnu_cxx::__pool<false>::_M_reclaim_block(char*, unsigned long)</code> (GLIBCXX_3.4.4) [LSB]
<code>__gnu_cxx::__pool<false>::_M_reserve_block(unsigned long, unsigned long)</code> (GLIBCXX_3.4.4) [LSB]

12.1.150 Class `__gnu_cxx::free_list`

12.1.150.1 Interfaces for Class `__gnu_cxx::free_list`

An LSB conforming implementation shall provide the architecture specific methods for Class `__gnu_cxx::free_list` specified in [Table 12-237](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-237 libstdcxx - Class `__gnu_cxx::free_list` Function Interfaces

<code>__gnu_cxx::free_list::_M_get(unsigned long)</code> (GLIBCXX_3.4.4) [LSB]
--

12.1.151 Class locale::_Impl

12.1.151.1 Interfaces for Class locale::_Impl

An LSB conforming implementation shall provide the architecture specific methods for Class std::locale::_Impl specified in [Table 12-238](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-238 libstdcxx - Class locale::_Impl Function Interfaces

locale::_Impl::_M_install_cache(locale::facet const*, unsigned long) (GLIBCXX_3.4.7) [ISOCXX]
--

12.1.152 Namespace std Functions

12.1.152.1 Interfaces for Namespace std Functions

An LSB conforming implementation shall provide the architecture specific methods for Namespace std Functions specified in [Table 12-239](#), with the full mandatory functionality as described in the referenced underlying specification.

Table 12-239 libstdcxx - Namespace std Functions Function Interfaces

long __copy_streambufs<char, char_traits<char>>>(basic_streambuf<char, char_traits<char>>*, basic_streambuf<char, char_traits<char>>*) (GLIBCXX_3.4.8) [ISOCXX]
--

long __copy_streambufs<wchar_t, char_traits<wchar_t>>>(basic_streambuf<wchar_t, char_traits<wchar_t>>*, basic_streambuf<wchar_t, char_traits<wchar_t>>*) (GLIBCXX_3.4.8) [ISOCXX]
--

12.1.153 Class char_traits<char>

12.1.153.1 Interfaces for Class char_traits<char>

No external methods are defined for libstdcxx - Class std::char_traits<char> in this part of the specification. See also the generic specification.

12.1.154 Class char_traits<wchar_t>

12.1.154.1 Interfaces for Class char_traits<wchar_t>

No external methods are defined for libstdcxx - Class std::char_traits<wchar_t> in this part of the specification. See also the generic specification.

12.2 Interface Definitions for libstdcxx

The interfaces defined on the following pages are included in libstdcxx and are defined by this specification. Unless otherwise noted, these interfaces shall be included in the source standard.

Other interfaces listed in [Section 12.1](#) shall behave as described in the referenced base document. For interfaces referencing LSB and not listed below, please see the generic part of the specification.

VI Package Format and Installation

13 Software Installation

13.1 Package Dependencies

The LSB runtime environment shall provide the following dependencies.

`lsb-core-amd64`

This dependency is used to indicate that the application is dependent on features contained in the LSB-Core specification.

This dependency shall have a version of 5.0.

Other LSB modules may add additional dependencies; such dependencies shall have the format `lsb-module-amd64`.

13.2 Package Architecture Considerations

All packages must specify an architecture of `x86_64`. An LSB runtime environment must accept an architecture of `x86_64` even if the native architecture is different.

The `archnum` value in the Lead Section shall be `0x0001`.

Annex A Alphabetical Listing of Interfaces by Library

A.1 libc

The behavior of the interfaces in this library is specified by the following Standards.

[Large File Support](#) [LFS]

[LSB Core - Generic](#) [LSB]

[RFC 5531/4506 RPC & XDR](#) [RPC + XDR]

[SUSv2](#) [SUSv2]

[POSIX 1003.1-2001 \(ISO/IEC 9945-2003\)](#) [SUSv3]

[POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#) [SUSv4]

[SVID Issue 4](#) [SVID.4]

Table A-1 libc Function Interfaces

_Exit (GLIBC_2.2.5) [SUSv4]	getopt (GLIBC_2.2.5) [LSB]	setbuf (GLIBC_2.2.5) [SUSv4]
_IO_feof (GLIBC_2.2.5) [LSB]	getopt_long (GLIBC_2.2.5) [LSB]	setbuffer (GLIBC_2.2.5) [LSB]
_IO_getc (GLIBC_2.2.5) [LSB]	getopt_long_only (GLIBC_2.2.5) [LSB]	setcontext (GLIBC_2.2.5) [SUSv3]
_IO_putc (GLIBC_2.2.5) [LSB]	getpagesize (GLIBC_2.2.5) [LSB]	setegid (GLIBC_2.2.5) [SUSv4]
_IO_puts (GLIBC_2.2.5) [LSB]	getpeername (GLIBC_2.2.5) [SUSv4]	setenv (GLIBC_2.2.5) [SUSv4]
__assert_fail (GLIBC_2.2.5) [LSB]	getpgid (GLIBC_2.2.5) [SUSv4]	seteuid (GLIBC_2.2.5) [SUSv4]
__ctype_get_mb_cur_max (GLIBC_2.2.5) [LSB]	getpgrp (GLIBC_2.2.5) [SUSv4]	setgid (GLIBC_2.2.5) [SUSv4]
__cxa_atexit (GLIBC_2.2.5) [LSB]	getpid (GLIBC_2.2.5) [SUSv4]	setgrent (GLIBC_2.2.5) [SUSv4]
__cxa_finalize (GLIBC_2.2.5) [LSB]	getppid (GLIBC_2.2.5) [SUSv4]	setgroups (GLIBC_2.2.5) [LSB]
__errno_location (GLIBC_2.2.5) [LSB]	getpriority (GLIBC_2.2.5) [SUSv4]	sethostname (GLIBC_2.2.5) [LSB]
__fpending (GLIBC_2.2.5) [LSB]	getprotobyname (GLIBC_2.2.5) [SUSv4]	setitimer (GLIBC_2.2.5) [SUSv4]
__fprintf_chk (GLIBC_2.3.4) [LSB]	getprotobyname_r (GLIBC_2.2.5) [LSB]	setlocale (GLIBC_2.2.5) [SUSv4]
__fxstat (GLIBC_2.2.5) [LSB]	getprotobynumber (GLIBC_2.2.5) [SUSv4]	setlogmask (GLIBC_2.2.5) [SUSv4]
__fxstat64 (GLIBC_2.2.5) [LSB]	getprotobynumber_r (GLIBC_2.2.5) [LSB]	setpgid (GLIBC_2.2.5) [SUSv4]
__getpagesize (GLIBC_2.2.5) [LSB]	getprotoent (GLIBC_2.2.5) [SUSv4]	setpgrp (GLIBC_2.2.5) [SUSv4]
__getpgid (GLIBC_2.2.5) [LSB]	getprotoent_r (GLIBC_2.2.5) [LSB]	setpriority (GLIBC_2.2.5) [SUSv4]
__h_errno_location (GLIBC_2.2.5) [LSB]	getpwent (GLIBC_2.2.5) [SUSv4]	setprotoent (GLIBC_2.2.5) [SUSv4]
__isinf (GLIBC_2.2.5) [LSB]	getpwent_r (GLIBC_2.2.5) [LSB]	setpwent (GLIBC_2.2.5) [SUSv4]
__isnff (GLIBC_2.2.5)	getpwnam (GLIBC_2.2.5)	setregid (GLIBC_2.2.5)

[LSB]	[SUSv4]	[SUSv4]
__isinfl(GLIBC_2.2.5) [LSB]	getpwnam_r(GLIBC_2.2.5) [SUSv4]	setreuid(GLIBC_2.2.5) [SUSv4]
__isnanc(GLIBC_2.2.5) [LSB]	getpwuid(GLIBC_2.2.5) [SUSv4]	setrlimit(GLIBC_2.2.5) [LSB]
__isnancf(GLIBC_2.2.5) [LSB]	getpwuid_r(GLIBC_2.2.5) [SUSv4]	setrlimit64(GLIBC_2.2.5) [LFS]
__isnancf_l(GLIBC_2.2.5) [LSB]	getrlimit(GLIBC_2.2.5) [LSB]	setservent(GLIBC_2.2.5) [SUSv4]
__libc_current_sigrtmax(GLIBC_2.2.5) [LSB]	getrlimit64(GLIBC_2.2.5) [LFS]	setuid(GLIBC_2.2.5) [SUSv4]
__libc_current_sigrtmin(GLIBC_2.2.5) [LSB]	getrusage(GLIBC_2.2.5) [SUSv4]	setsockopt(GLIBC_2.2.5) [LSB]
__libc_start_main(GLIBC_2.2.5) [LSB]	getservbyname(GLIBC_2.2.5) [SUSv4]	setstate(GLIBC_2.2.5) [SUSv4]
__lxstat(GLIBC_2.2.5) [LSB]	getservbyname_r(GLIBC_2.2.5) [LSB]	setstate_r(GLIBC_2.2.5) [LSB]
__lxstat64(GLIBC_2.2.5) [LSB]	getservbyport(GLIBC_2.2.5) [SUSv4]	setuid(GLIBC_2.2.5) [SUSv4]
__memcpy(GLIBC_2.2.5) [LSB]	getservbyport_r(GLIBC_2.2.5) [LSB]	setutent(GLIBC_2.2.5) [LSB]
__printf_chk(GLIBC_2.3.4) [LSB]	setservent(GLIBC_2.2.5) [SUSv4]	setutxent(GLIBC_2.2.5) [SUSv4]
__rawmemchr(GLIBC_2.2.5) [LSB]	setservent_r(GLIBC_2.2.5) [LSB]	setvbuf(GLIBC_2.2.5) [SUSv4]
__sigsetjmp(GLIBC_2.2.5) [LSB]	getsid(GLIBC_2.2.5) [SUSv4]	shmat(GLIBC_2.2.5) [SUSv4]
__snprintf_chk(GLIBC_2.3.4) [LSB]	getsockname(GLIBC_2.2.5) [SUSv4]	shmctl(GLIBC_2.2.5) [SUSv4]
__sprintf_chk(GLIBC_2.3.4) [LSB]	getsockopt(GLIBC_2.2.5) [LSB]	shmdt(GLIBC_2.2.5) [SUSv4]
__stpcpy(GLIBC_2.2.5) [LSB]	getsubopt(GLIBC_2.2.5) [SUSv4]	shmget(GLIBC_2.2.5) [SUSv4]
__strdup(GLIBC_2.2.5) [LSB]	gettext(GLIBC_2.2.5) [LSB]	shutdown(GLIBC_2.2.5) [SUSv4]
__strtod_internal(GLIBC_2.2.5) [LSB]	gettimeofday(GLIBC_2.2.5) [SUSv4]	sigaction(GLIBC_2.2.5) [SUSv4]
__strtof_internal(GLIBC_2.2.5) [LSB]	getuid(GLIBC_2.2.5) [SUSv4]	sigaddset(GLIBC_2.2.5) [SUSv4]
__strtok_r(GLIBC_2.2.5) [LSB]	getutent(GLIBC_2.2.5) [LSB]	sigaltstack(GLIBC_2.2.5) [SUSv4]
__strtol_internal(GLIBC_2.2.5) [LSB]	getutent_r(GLIBC_2.2.5) [LSB]	sigandset(GLIBC_2.2.5) [LSB]
__strtold_internal(GLIBC_2.2.5) [LSB]	getutxent(GLIBC_2.2.5) [SUSv4]	sigdelset(GLIBC_2.2.5) [SUSv4]
__strtoll_internal(GLIBC_2.2.5) [LSB]	getutxid(GLIBC_2.2.5) [SUSv4]	sigemptyset(GLIBC_2.2.5) [SUSv4]
__strtoul_internal(GLIBC_2.2.5) [LSB]	getutxline(GLIBC_2.2.5) [SUSv4]	sigfillset(GLIBC_2.2.5) [SUSv4]

LSB Core - X86-64 5.0

__strtoull_internal(GLIBC_2.2.5) [LSB]	getw(GLIBC_2.2.5) [SUSv2]	sighold(GLIBC_2.2.5) [SUSv4]
__sysconf(GLIBC_2.2.5) [LSB]	getwc(GLIBC_2.2.5) [SUSv4]	sigignore(GLIBC_2.2.5) [SUSv4]
__sysv_signal(GLIBC_2.2.5) [LSB]	getwc_unlocked(GLIBC_2.2.5) [LSB]	siginterrupt(GLIBC_2.2.5) [SUSv4]
__vfprintf_chk(GLIBC_2.3.4) [LSB]	getwchar(GLIBC_2.2.5) [SUSv4]	sigisemptyset(GLIBC_2.2.5) [LSB]
__vprintf_chk(GLIBC_2.3.4) [LSB]	getwchar_unlocked(GLIBC_2.2.5) [LSB]	sigismember(GLIBC_2.2.5) [SUSv4]
__vsnprintf_chk(GLIBC_2.3.4) [LSB]	getwd(GLIBC_2.2.5) [SUSv3]	siglongjmp(GLIBC_2.2.5) [SUSv4]
__vsprintf_chk(GLIBC_2.3.4) [LSB]	glob(GLIBC_2.2.5) [SUSv4]	signal(GLIBC_2.2.5) [SUSv4]
__wcstod_internal(GLIBC_2.2.5) [LSB]	glob64(GLIBC_2.2.5) [LSB]	sigorset(GLIBC_2.2.5) [LSB]
__wcstof_internal(GLIBC_2.2.5) [LSB]	globfree(GLIBC_2.2.5) [SUSv4]	sigpause(GLIBC_2.2.5) [LSB]
__wcstol_internal(GLIBC_2.2.5) [LSB]	globfree64(GLIBC_2.2.5) [LSB]	sigpending(GLIBC_2.2.5) [SUSv4]
__wcstold_internal(GLIBC_2.2.5) [LSB]	gmtime(GLIBC_2.2.5) [SUSv4]	sigprocmask(GLIBC_2.2.5) [SUSv4]
__wcstoul_internal(GLIBC_2.2.5) [LSB]	gmtime_r(GLIBC_2.2.5) [SUSv4]	sigqueue(GLIBC_2.2.5) [SUSv4]
__xmknod(GLIBC_2.2.5) [LSB]	gnu_get_libc_release(GLIBC_2.2.5) [LSB]	sigrelse(GLIBC_2.2.5) [SUSv4]
__xpg_basename(GLIBC_2.2.5) [LSB]	gnu_get_libc_version(GLIBC_2.2.5) [LSB]	sigreturn(GLIBC_2.2.5) [LSB]
__xpg_sigpause(GLIBC_2.2.5) [LSB]	grantpt(GLIBC_2.2.5) [SUSv4]	sigset(GLIBC_2.2.5) [SUSv4]
__xpg_strerror_r(GLIBC_2.3.4) [LSB]	hcreate(GLIBC_2.2.5) [SUSv4]	sigsuspend(GLIBC_2.2.5) [SUSv4]
__xstat(GLIBC_2.2.5) [LSB]	hcreate_r(GLIBC_2.2.5) [LSB]	sigtimedwait(GLIBC_2.2.5) [SUSv4]
__xstat64(GLIBC_2.2.5) [LSB]	hdestroy(GLIBC_2.2.5) [SUSv4]	sigwait(GLIBC_2.2.5) [SUSv4]
_exit(GLIBC_2.2.5) [SUSv4]	hdestroy_r(GLIBC_2.2.5) [LSB]	sigwaitinfo(GLIBC_2.2.5) [SUSv4]
_longjmp(GLIBC_2.2.5) [SUSv4]	hsearch(GLIBC_2.2.5) [SUSv4]	sleep(GLIBC_2.2.5) [SUSv4]
_setjmp(GLIBC_2.2.5) [SUSv4]	hsearch_r(GLIBC_2.2.5) [LSB]	snprintf(GLIBC_2.2.5) [SUSv4]
_tolower(GLIBC_2.2.5) [SUSv4]	htonl(GLIBC_2.2.5) [SUSv4]	socketatmark(GLIBC_2.2.5) [SUSv4]
_toupper(GLIBC_2.2.5) [SUSv4]	htons(GLIBC_2.2.5) [SUSv4]	socket(GLIBC_2.2.5) [SUSv4]
a64l(GLIBC_2.2.5) [SUSv4]	iconv(GLIBC_2.2.5) [SUSv4]	socketpair(GLIBC_2.2.5) [SUSv4]
abort(GLIBC_2.2.5) [SUSv4]	iconv_close(GLIBC_2.2.5) [SUSv4]	sprintf(GLIBC_2.2.5) [SUSv4]

abs(GLIBC_2.2.5) [SUSv4]	iconv_open(GLIBC_2.2.5) [SUSv4]	srand(GLIBC_2.2.5) [SUSv4]
accept(GLIBC_2.2.5) [SUSv4]	if_freenameindex(GLIBC_2.2.5) [SUSv4]	srand48(GLIBC_2.2.5) [SUSv4]
access(GLIBC_2.2.5) [SUSv4]	if_indextname(GLIBC_2.2.5) [SUSv4]	srand48_r(GLIBC_2.2.5) [LSB]
acct(GLIBC_2.2.5) [LSB]	if_nameindex(GLIBC_2.2.5) [SUSv4]	srandom(GLIBC_2.2.5) [SUSv4]
adjtime(GLIBC_2.2.5) [LSB]	if_nametoindex(GLIBC_2.2.5) [SUSv4]	srandom_r(GLIBC_2.2.5) [LSB]
alarm(GLIBC_2.2.5) [SUSv4]	imaxabs(GLIBC_2.2.5) [SUSv4]	sscanf(GLIBC_2.2.5) [LSB]
alphasort(GLIBC_2.2.5) [SUSv4]	imaxdiv(GLIBC_2.2.5) [SUSv4]	statfs(GLIBC_2.2.5) [LSB]
alphasort64(GLIBC_2.2.5) [LSB]	index(GLIBC_2.2.5) [SUSv3]	statfs64(GLIBC_2.2.5) [LSB]
argz_add(GLIBC_2.2.5) [LSB]	inet_addr(GLIBC_2.2.5) [SUSv4]	statvfs(GLIBC_2.2.5) [SUSv4]
argz_add_sep(GLIBC_2.2.5) [LSB]	inet_aton(GLIBC_2.2.5) [LSB]	statvfs64(GLIBC_2.2.5) [LFS]
argz_append(GLIBC_2.2.5) [LSB]	inet_ntoa(GLIBC_2.2.5) [SUSv4]	stime(GLIBC_2.2.5) [LSB]
argz_count(GLIBC_2.2.5) [LSB]	inet_ntop(GLIBC_2.2.5) [SUSv4]	stpncpy(GLIBC_2.2.5) [SUSv4]
argz_create(GLIBC_2.2.5) [LSB]	inet_pton(GLIBC_2.2.5) [SUSv4]	stpncpy(GLIBC_2.2.5) [SUSv4]
argz_create_sep(GLIBC_2.2.5) [LSB]	initgroups(GLIBC_2.2.5) [LSB]	strcasecmp(GLIBC_2.2.5) [SUSv4]
argz_delete(GLIBC_2.2.5) [LSB]	initstate(GLIBC_2.2.5) [SUSv4]	strcasestr(GLIBC_2.2.5) [LSB]
argz_extract(GLIBC_2.2.5) [LSB]	initstate_r(GLIBC_2.2.5) [LSB]	strcat(GLIBC_2.2.5) [SUSv4]
argz_insert(GLIBC_2.2.5) [LSB]	insque(GLIBC_2.2.5) [SUSv4]	strchr(GLIBC_2.2.5) [SUSv4]
argz_next(GLIBC_2.2.5) [LSB]	ioctl(GLIBC_2.2.5) [LSB]	strcmp(GLIBC_2.2.5) [SUSv4]
argz_replace(GLIBC_2.2.5) [LSB]	ioperm(GLIBC_2.2.5) [LSB]	strcoll(GLIBC_2.2.5) [SUSv4]
argz_stringify(GLIBC_2.2.5) [LSB]	iopl(GLIBC_2.2.5) [LSB]	strcpy(GLIBC_2.2.5) [SUSv4]
asctime(GLIBC_2.2.5) [SUSv4]	isalnum(GLIBC_2.2.5) [SUSv4]	strcspn(GLIBC_2.2.5) [SUSv4]
asctime_r(GLIBC_2.2.5) [SUSv4]	isalpha(GLIBC_2.2.5) [SUSv4]	strdup(GLIBC_2.2.5) [SUSv4]
asprintf(GLIBC_2.2.5) [LSB]	isascii(GLIBC_2.2.5) [SUSv4]	strerror(GLIBC_2.2.5) [SUSv4]
atof(GLIBC_2.2.5) [SUSv4]	isatty(GLIBC_2.2.5) [SUSv4]	strerror_r(GLIBC_2.2.5) [LSB]
atoi(GLIBC_2.2.5) [SUSv4]	isblank(GLIBC_2.2.5) [SUSv4]	strfmon(GLIBC_2.2.5) [SUSv4]

LSB Core - X86-64 5.0

atol(GLIBC_2.2.5) [SUSv4]	isctrl(GLIBC_2.2.5) [SUSv4]	strftime(GLIBC_2.2.5) [SUSv4]
atoll(GLIBC_2.2.5) [SUSv4]	isdigit(GLIBC_2.2.5) [SUSv4]	strlen(GLIBC_2.2.5) [SUSv4]
authnone_create(GLIBC_2.2.5) [SVID.4]	isgraph(GLIBC_2.2.5) [SUSv4]	strncasecmp(GLIBC_2.2.5) [SUSv4]
backtrace(GLIBC_2.2.5) [LSB]	islower(GLIBC_2.2.5) [SUSv4]	strncat(GLIBC_2.2.5) [SUSv4]
backtrace_symbols(GLIBC_2.2.5) [LSB]	isprint(GLIBC_2.2.5) [SUSv4]	strncmp(GLIBC_2.2.5) [SUSv4]
backtrace_symbols_fd(GLIBC_2.2.5) [LSB]	ispunct(GLIBC_2.2.5) [SUSv4]	strncpy(GLIBC_2.2.5) [SUSv4]
basename(GLIBC_2.2.5) [LSB]	isspace(GLIBC_2.2.5) [SUSv4]	strndup(GLIBC_2.2.5) [SUSv4]
bcmp(GLIBC_2.2.5) [SUSv3]	isupper(GLIBC_2.2.5) [SUSv4]	strnlen(GLIBC_2.2.5) [SUSv4]
bcopy(GLIBC_2.2.5) [SUSv3]	iswalnum(GLIBC_2.2.5) [SUSv4]	strpbrk(GLIBC_2.2.5) [SUSv4]
bind(GLIBC_2.2.5) [SUSv4]	iswalpha(GLIBC_2.2.5) [SUSv4]	strptime(GLIBC_2.2.5) [LSB]
bind_textdomain_codeset(GLIBC_2.2.5) [LSB]	iswblank(GLIBC_2.2.5) [SUSv4]	strrchr(GLIBC_2.2.5) [SUSv4]
bindresvport(GLIBC_2.2.5) [LSB]	iswcntrl(GLIBC_2.2.5) [SUSv4]	strsep(GLIBC_2.2.5) [LSB]
bindtextdomain(GLIBC_2.2.5) [LSB]	iswctype(GLIBC_2.2.5) [SUSv4]	strsignal(GLIBC_2.2.5) [SUSv4]
brk(GLIBC_2.2.5) [SUSv2]	iswdigit(GLIBC_2.2.5) [SUSv4]	strspn(GLIBC_2.2.5) [SUSv4]
bsd_signal(GLIBC_2.2.5) [SUSv3]	iswgraph(GLIBC_2.2.5) [SUSv4]	strstr(GLIBC_2.2.5) [SUSv4]
bsearch(GLIBC_2.2.5) [SUSv4]	iswlower(GLIBC_2.2.5) [SUSv4]	strtod(GLIBC_2.2.5) [SUSv4]
btowc(GLIBC_2.2.5) [SUSv4]	iswprint(GLIBC_2.2.5) [SUSv4]	strtof(GLIBC_2.2.5) [SUSv4]
bzero(GLIBC_2.2.5) [SUSv3]	iswpunct(GLIBC_2.2.5) [SUSv4]	strtoimax(GLIBC_2.2.5) [SUSv4]
calloc(GLIBC_2.2.5) [SUSv4]	iswspace(GLIBC_2.2.5) [SUSv4]	strtok(GLIBC_2.2.5) [SUSv4]
callrpc(GLIBC_2.2.5) [RPC + XDR]	iswupper(GLIBC_2.2.5) [SUSv4]	strtok_r(GLIBC_2.2.5) [SUSv4]
catclose(GLIBC_2.2.5) [SUSv4]	iswxdigit(GLIBC_2.2.5) [SUSv4]	strtol(GLIBC_2.2.5) [SUSv4]
catgets(GLIBC_2.2.5) [SUSv4]	isxdigit(GLIBC_2.2.5) [SUSv4]	strtold(GLIBC_2.2.5) [SUSv4]
catopen(GLIBC_2.2.5) [SUSv4]	jrand48(GLIBC_2.2.5) [SUSv4]	strtoll(GLIBC_2.2.5) [SUSv4]
cfgetispeed(GLIBC_2.2.5) [SUSv4]	jrand48_r(GLIBC_2.2.5) [LSB]	strtoq(GLIBC_2.2.5) [LSB]
cfgetospeed(GLIBC_2.2.5) [SUSv4]	key_decryptsession(GLIBC_2.2.5) [SVID.4]	strtoul(GLIBC_2.2.5) [SUSv4]

cfmakeraw(GLIBC_2.2.5) [LSB]	kill(GLIBC_2.2.5) [LSB]	strtoull(GLIBC_2.2.5) [SUSv4]
cfsetispeed(GLIBC_2.2.5) [SUSv4]	killpg(GLIBC_2.2.5) [SUSv4]	strtoumax(GLIBC_2.2.5) [SUSv4]
cfsetospeed(GLIBC_2.2.5) [SUSv4]	l64a(GLIBC_2.2.5) [SUSv4]	strtouq(GLIBC_2.2.5) [LSB]
cfsetspeed(GLIBC_2.2.5) [LSB]	labs(GLIBC_2.2.5) [SUSv4]	strxfrm(GLIBC_2.2.5) [SUSv4]
chdir(GLIBC_2.2.5) [SUSv4]	lchown(GLIBC_2.2.5) [SUSv4]	svc_getreqset(GLIBC_2.2.5) [SVID.4]
chmod(GLIBC_2.2.5) [SUSv4]	lcong48(GLIBC_2.2.5) [SUSv4]	svc_register(GLIBC_2.2.5) [LSB]
chown(GLIBC_2.2.5) [SUSv4]	lcong48_r(GLIBC_2.2.5) [LSB]	svc_run(GLIBC_2.2.5) [LSB]
chroot(GLIBC_2.2.5) [SUSv2]	ldiv(GLIBC_2.2.5) [SUSv4]	svc_sendreply(GLIBC_2.2.5) [LSB]
clearerr(GLIBC_2.2.5) [SUSv4]	lfind(GLIBC_2.2.5) [SUSv4]	svcerr_auth(GLIBC_2.2.5) [SVID.4]
clearerr_unlocked(GLIBC_2.2.5) [LSB]	link(GLIBC_2.2.5) [LSB]	svcerr_decode(GLIBC_2.2.5) [SVID.4]
clnt_create(GLIBC_2.2.5) [SVID.4]	listen(GLIBC_2.2.5) [SUSv4]	svcerr_noproc(GLIBC_2.2.5) [SVID.4]
clnt_pcreateerror(GLIBC_2.2.5) [SVID.4]	llabs(GLIBC_2.2.5) [SUSv4]	svcerr_noprogram(GLIBC_2.2.5) [SVID.4]
clnt_permpno(GLIBC_2.2.5) [SVID.4]	lldiv(GLIBC_2.2.5) [SUSv4]	svcerr_progvers(GLIBC_2.2.5) [SVID.4]
clnt_perror(GLIBC_2.2.5) [SVID.4]	localeconv(GLIBC_2.2.5) [SUSv4]	svcerr_systemerr(GLIBC_2.2.5) [SVID.4]
clnt_spccreateerror(GLIBC_2.2.5) [SVID.4]	localtime(GLIBC_2.2.5) [SUSv4]	svcerr_weakauth(GLIBC_2.2.5) [SVID.4]
clnt_sperrno(GLIBC_2.2.5) [SVID.4]	localtime_r(GLIBC_2.2.5) [SUSv4]	svcfld_create(GLIBC_2.2.5) [RPC + XDR]
clnt_spperror(GLIBC_2.2.5) [SVID.4]	lockf(GLIBC_2.2.5) [SUSv4]	svcraw_create(GLIBC_2.2.5) [RPC + XDR]
clntraw_create(GLIBC_2.2.5) [RPC + XDR]	lockf64(GLIBC_2.2.5) [LFS]	svctcp_create(GLIBC_2.2.5) [LSB]
clnttcp_create(GLIBC_2.2.5) [RPC + XDR]	longjmp(GLIBC_2.2.5) [SUSv4]	svcudp_create(GLIBC_2.2.5) [LSB]
clntudp_bufcreate(GLIBC_2.2.5) [RPC + XDR]	lrand48(GLIBC_2.2.5) [SUSv4]	swab(GLIBC_2.2.5) [SUSv4]
clntudp_create(GLIBC_2.2.5) [RPC + XDR]	lrand48_r(GLIBC_2.2.5) [LSB]	swapcontext(GLIBC_2.2.5) [SUSv3]
clock(GLIBC_2.2.5) [SUSv4]	lsearch(GLIBC_2.2.5) [SUSv4]	swprintf(GLIBC_2.2.5) [SUSv4]
close(GLIBC_2.2.5) [SUSv4]	lseek(GLIBC_2.2.5) [SUSv4]	swscanf(GLIBC_2.2.5) [LSB]
closedir(GLIBC_2.2.5) [SUSv4]	lseek64(GLIBC_2.2.5) [LFS]	symlink(GLIBC_2.2.5) [SUSv4]
closelog(GLIBC_2.2.5) [SUSv4]	makecontext(GLIBC_2.2.5) [SUSv3]	sync(GLIBC_2.2.5) [SUSv4]

LSB Core - X86-64 5.0

confstr(GLIBC_2.2.5) [SUSv4]	malloc(GLIBC_2.2.5) [SUSv4]	sysconf(GLIBC_2.2.5) [LSB]
connect(GLIBC_2.2.5) [SUSv4]	mblen(GLIBC_2.2.5) [SUSv4]	sysinfo(GLIBC_2.2.5) [LSB]
creat(GLIBC_2.2.5) [SUSv4]	mbrlen(GLIBC_2.2.5) [SUSv4]	syslog(GLIBC_2.2.5) [SUSv4]
creat64(GLIBC_2.2.5) [LFS]	mbtowc(GLIBC_2.2.5) [SUSv4]	system(GLIBC_2.2.5) [LSB]
ctermid(GLIBC_2.2.5) [SUSv4]	mbsinit(GLIBC_2.2.5) [SUSv4]	tcdrain(GLIBC_2.2.5) [SUSv4]
ctime(GLIBC_2.2.5) [SUSv4]	mbsnrtowcs(GLIBC_2.2.5) [SUSv4]	tcflow(GLIBC_2.2.5) [SUSv4]
ctime_r(GLIBC_2.2.5) [SUSv4]	mbsrtowcs(GLIBC_2.2.5) [SUSv4]	tcflush(GLIBC_2.2.5) [SUSv4]
cuserid(GLIBC_2.2.5) [SUSv2]	mbstowcs(GLIBC_2.2.5) [SUSv4]	tcgetattr(GLIBC_2.2.5) [SUSv4]
daemon(GLIBC_2.2.5) [LSB]	mbtowc(GLIBC_2.2.5) [SUSv4]	tcgetpgrp(GLIBC_2.2.5) [SUSv4]
dcgettext(GLIBC_2.2.5) [LSB]	memccpy(GLIBC_2.2.5) [SUSv4]	tcgetsid(GLIBC_2.2.5) [SUSv4]
dcngettext(GLIBC_2.2.5) [LSB]	memchr(GLIBC_2.2.5) [SUSv4]	tcsendbreak(GLIBC_2.2.5) [SUSv4]
dgettext(GLIBC_2.2.5) [LSB]	memcmp(GLIBC_2.2.5) [SUSv4]	tcsetattr(GLIBC_2.2.5) [SUSv4]
difftime(GLIBC_2.2.5) [SUSv4]	memcpy(GLIBC_2.14) [SUSv4]	tcsetpgrp(GLIBC_2.2.5) [SUSv4]
dirfd(GLIBC_2.2.5) [SUSv4]	memmem(GLIBC_2.2.5) [LSB]	tdelete(GLIBC_2.2.5) [SUSv4]
dirname(GLIBC_2.2.5) [SUSv4]	memmove(GLIBC_2.2.5) [SUSv4]	telldir(GLIBC_2.2.5) [SUSv4]
div(GLIBC_2.2.5) [SUSv4]	memrchr(GLIBC_2.2.5) [LSB]	tempnam(GLIBC_2.2.5) [SUSv4]
dl_iterate_phdr(GLIBC_2.2.5) [LSB]	memset(GLIBC_2.2.5) [SUSv4]	textdomain(GLIBC_2.2.5) [LSB]
dngettext(GLIBC_2.2.5) [LSB]	mkdir(GLIBC_2.2.5) [SUSv4]	tfind(GLIBC_2.2.5) [SUSv4]
dprintf(GLIBC_2.2.5) [SUSv4]	mkdtemp(GLIBC_2.2.5) [SUSv4]	time(GLIBC_2.2.5) [SUSv4]
drand48(GLIBC_2.2.5) [SUSv4]	mkfifo(GLIBC_2.2.5) [SUSv4]	times(GLIBC_2.2.5) [SUSv4]
drand48_r(GLIBC_2.2.5) [LSB]	mkstemp(GLIBC_2.2.5) [SUSv4]	tmpfile(GLIBC_2.2.5) [SUSv4]
dup(GLIBC_2.2.5) [SUSv4]	mkstemp64(GLIBC_2.2.5) [LSB]	tmpfile64(GLIBC_2.2.5) [LFS]
dup2(GLIBC_2.2.5) [SUSv4]	mktemp(GLIBC_2.2.5) [SUSv3]	tmpnam(GLIBC_2.2.5) [SUSv4]
ecvt(GLIBC_2.2.5) [SUSv3]	mktime(GLIBC_2.2.5) [SUSv4]	toascii(GLIBC_2.2.5) [SUSv4]
endgrent(GLIBC_2.2.5) [SUSv4]	mlock(GLIBC_2.2.5) [SUSv4]	tolower(GLIBC_2.2.5) [SUSv4]

endprotoent(GLIBC_2.2.5) [SUSv4]	mlockall(GLIBC_2.2.5) [SUSv4]	toupper(GLIBC_2.2.5) [SUSv4]
endpwent(GLIBC_2.2.5) [SUSv4]	mmap(GLIBC_2.2.5) [SUSv4]	towctrans(GLIBC_2.2.5) [SUSv4]
endservent(GLIBC_2.2.5) [SUSv4]	mmap64(GLIBC_2.2.5) [LFS]	towlower(GLIBC_2.2.5) [SUSv4]
endutent(GLIBC_2.2.5) [LSB]	mprotect(GLIBC_2.2.5) [SUSv4]	towupper(GLIBC_2.2.5) [SUSv4]
endutxent(GLIBC_2.2.5) [SUSv4]	rand48(GLIBC_2.2.5) [SUSv4]	truncate(GLIBC_2.2.5) [SUSv4]
envz_add(GLIBC_2.2.5) [LSB]	rand48_r(GLIBC_2.2.5) [LSB]	truncate64(GLIBC_2.2.5) [LFS]
envz_entry(GLIBC_2.2.5) [LSB]	mremap(GLIBC_2.2.5) [LSB]	tsearch(GLIBC_2.2.5) [SUSv4]
envz_get(GLIBC_2.2.5) [LSB]	msgctl(GLIBC_2.2.5) [SUSv4]	ttynamename(GLIBC_2.2.5) [SUSv4]
envz_merge(GLIBC_2.2.5) [LSB]	msgget(GLIBC_2.2.5) [SUSv4]	ttynamename_r(GLIBC_2.2.5) [SUSv4]
envz_remove(GLIBC_2.2.5) [LSB]	msgrcv(GLIBC_2.2.5) [SUSv4]	twalk(GLIBC_2.2.5) [SUSv4]
envz_strip(GLIBC_2.2.5) [LSB]	msgsnd(GLIBC_2.2.5) [SUSv4]	tzset(GLIBC_2.2.5) [SUSv4]
erand48(GLIBC_2.2.5) [SUSv4]	msync(GLIBC_2.2.5) [SUSv4]	ualarm(GLIBC_2.2.5) [SUSv3]
erand48_r(GLIBC_2.2.5) [LSB]	munlock(GLIBC_2.2.5) [SUSv4]	ulimit(GLIBC_2.2.5) [SUSv4]
err(GLIBC_2.2.5) [LSB]	munlockall(GLIBC_2.2.5) [SUSv4]	umask(GLIBC_2.2.5) [SUSv4]
error(GLIBC_2.2.5) [LSB]	munmap(GLIBC_2.2.5) [SUSv4]	uname(GLIBC_2.2.5) [SUSv4]
errx(GLIBC_2.2.5) [LSB]	nanosleep(GLIBC_2.2.5) [SUSv4]	ungetc(GLIBC_2.2.5) [SUSv4]
execl(GLIBC_2.2.5) [SUSv4]	nftw(GLIBC_2.3.3) [SUSv4]	ungetwc(GLIBC_2.2.5) [SUSv4]
execle(GLIBC_2.2.5) [SUSv4]	nftw64(GLIBC_2.3.3) [LFS]	unlink(GLIBC_2.2.5) [LSB]
execlp(GLIBC_2.2.5) [SUSv4]	ngettext(GLIBC_2.2.5) [LSB]	unlockpt(GLIBC_2.2.5) [SUSv4]
execv(GLIBC_2.2.5) [SUSv4]	nice(GLIBC_2.2.5) [SUSv4]	unsetenv(GLIBC_2.2.5) [SUSv4]
execve(GLIBC_2.2.5) [SUSv4]	nl_langinfo(GLIBC_2.2.5) [SUSv4]	usleep(GLIBC_2.2.5) [SUSv3]
execvp(GLIBC_2.2.5) [SUSv4]	nrand48(GLIBC_2.2.5) [SUSv4]	utime(GLIBC_2.2.5) [SUSv4]
exit(GLIBC_2.2.5) [SUSv4]	nrand48_r(GLIBC_2.2.5) [LSB]	utimes(GLIBC_2.2.5) [SUSv4]
fchdir(GLIBC_2.2.5) [SUSv4]	ntohl(GLIBC_2.2.5) [SUSv4]	utmpname(GLIBC_2.2.5) [LSB]
fchmod(GLIBC_2.2.5) [SUSv4]	ntohs(GLIBC_2.2.5) [SUSv4]	vasprintf(GLIBC_2.2.5) [LSB]

LSB Core - X86-64 5.0

fchown(GLIBC_2.2.5) [SUSv4]	open(GLIBC_2.2.5) [SUSv4]	vdprintf(GLIBC_2.2.5) [SUSv4]
fclose(GLIBC_2.2.5) [SUSv4]	open64(GLIBC_2.2.5) [LFS]	verrx(GLIBC_2.2.5) [LSB]
fcntl(GLIBC_2.2.5)[LSB]	open_memstream(GLIBC_2.2.5)[SUSv4]	vfork(GLIBC_2.2.5) [SUSv3]
fcvt(GLIBC_2.2.5) [SUSv3]	opendir(GLIBC_2.2.5) [SUSv4]	vfprintf(GLIBC_2.2.5) [SUSv4]
fdatasync(GLIBC_2.2.5) [SUSv4]	openlog(GLIBC_2.2.5) [SUSv4]	vfscanf(GLIBC_2.2.5) [LSB]
fdopen(GLIBC_2.2.5) [SUSv4]	pathconf(GLIBC_2.2.5) [SUSv4]	vwprintf(GLIBC_2.2.5) [SUSv4]
feof(GLIBC_2.2.5) [SUSv4]	pause(GLIBC_2.2.5) [SUSv4]	vwscanf(GLIBC_2.2.5) [LSB]
feof_unlocked(GLIBC_2.2.5)[LSB]	pclose(GLIBC_2.2.5) [SUSv4]	vprintf(GLIBC_2.2.5) [SUSv4]
ferror(GLIBC_2.2.5) [SUSv4]	perror(GLIBC_2.2.5) [SUSv4]	vscanf(GLIBC_2.2.5) [LSB]
ferror_unlocked(GLIBC_2.2.5)[LSB]	pipe(GLIBC_2.2.5) [SUSv4]	vsprintf(GLIBC_2.2.5) [SUSv4]
fexecve(GLIBC_2.2.5) [SUSv4]	pmap_getport(GLIBC_2.2.5)[LSB]	vsprintf(GLIBC_2.2.5) [SUSv4]
fflush(GLIBC_2.2.5) [SUSv4]	pmap_set(GLIBC_2.2.5) [LSB]	vsscanf(GLIBC_2.2.5) [LSB]
fflush_unlocked(GLIBC_2.2.5)[LSB]	pmap_unset(GLIBC_2.2.5)[LSB]	vswprintf(GLIBC_2.2.5) [SUSv4]
ffs(GLIBC_2.2.5) [SUSv4]	poll(GLIBC_2.2.5) [SUSv4]	vswscanf(GLIBC_2.2.5) [LSB]
fgetc(GLIBC_2.2.5) [SUSv4]	popen(GLIBC_2.2.5) [SUSv4]	vsyslog(GLIBC_2.2.5) [LSB]
fgetc_unlocked(GLIBC_2.2.5)[LSB]	posix_fadvise(GLIBC_2.2.5)[SUSv4]	vwprintf(GLIBC_2.2.5) [SUSv4]
fgetpos(GLIBC_2.2.5) [SUSv4]	posix_fadvise64(GLIBC_2.2.5)[LSB]	vwscanf(GLIBC_2.2.5) [LSB]
fgetpos64(GLIBC_2.2.5) [LFS]	posix_fallocate(GLIBC_2.2.5)[SUSv4]	wait(GLIBC_2.2.5) [SUSv4]
fgets(GLIBC_2.2.5) [SUSv4]	posix_fallocate64(GLIBC_2.2.5)[LSB]	wait4(GLIBC_2.2.5) [LSB]
fgets_unlocked(GLIBC_2.2.5)[LSB]	posix_madvise(GLIBC_2.2.5)[SUSv4]	waitid(GLIBC_2.2.5) [SUSv4]
fgetwc(GLIBC_2.2.5) [SUSv4]	posix_memalign(GLIBC_2.2.5)[SUSv4]	waitpid(GLIBC_2.2.5) [SUSv4]
fgetwc_unlocked(GLIBC_2.2.5)[LSB]	posix_openpt(GLIBC_2.2.5)[SUSv4]	warn(GLIBC_2.2.5) [LSB]
fgetws(GLIBC_2.2.5) [SUSv4]	posix_spawn(GLIBC_2.1.5)[SUSv4]	warnx(GLIBC_2.2.5) [LSB]
fgetws_unlocked(GLIBC_2.2.5)[LSB]	posix_spawn_file_actions_addclose(GLIBC_2.2.5) [SUSv4]	wcpcpy(GLIBC_2.2.5) [SUSv4]
fileno(GLIBC_2.2.5)	posix_spawn_file_actions	wcpncpy(GLIBC_2.2.5)

[SUSv4]	<code>_adddup2</code> (GLIBC_2.2.5) [SUSv4]	[SUSv4]
<code>fileno_unlocked</code> (GLIBC_2.2.5) [LSB]	<code>posix_spawn_file_actions_addopen</code> (GLIBC_2.2.5) [SUSv4]	<code>wcrtomb</code> (GLIBC_2.2.5) [SUSv4]
<code>flock</code> (GLIBC_2.2.5) [LSB]	<code>posix_spawn_file_actions_destroy</code> (GLIBC_2.2.5) [SUSv4]	<code>wcscasecmp</code> (GLIBC_2.2.5) [SUSv4]
<code>flockfile</code> (GLIBC_2.2.5) [SUSv4]	<code>posix_spawn_file_actions_init</code> (GLIBC_2.2.5) [SUSv4]	<code>wcscat</code> (GLIBC_2.2.5) [SUSv4]
<code>fmemopen</code> (GLIBC_2.2.5) [SUSv4]	<code>posix_spawnattr_destroy</code> (GLIBC_2.2.5) [SUSv4]	<code>wcschr</code> (GLIBC_2.2.5) [SUSv4]
<code>fmsg</code> (GLIBC_2.2.5) [SUSv4]	<code>posix_spawnattr_getflags</code> (GLIBC_2.2.5) [SUSv4]	<code>wcscmp</code> (GLIBC_2.2.5) [SUSv4]
<code>fnmatch</code> (GLIBC_2.2.5) [LSB]	<code>posix_spawnattr_getpgroup</code> (GLIBC_2.2.5) [SUSv4]	<code>wscoll</code> (GLIBC_2.2.5) [SUSv4]
<code>fopen</code> (GLIBC_2.2.5) [SUSv4]	<code>posix_spawnattr_getschedparam</code> (GLIBC_2.2.5) [SUSv4]	<code>wscpy</code> (GLIBC_2.2.5) [SUSv4]
<code>fopen64</code> (GLIBC_2.2.5) [LFS]	<code>posix_spawnattr_getschedpolicy</code> (GLIBC_2.2.5) [SUSv4]	<code>wscspn</code> (GLIBC_2.2.5) [SUSv4]
<code>fork</code> (GLIBC_2.2.5) [SUSv4]	<code>posix_spawnattr_getsigdefault</code> (GLIBC_2.2.5) [SUSv4]	<code>wcsdup</code> (GLIBC_2.2.5) [SUSv4]
<code>fpathconf</code> (GLIBC_2.2.5) [SUSv4]	<code>posix_spawnattr_getsigmask</code> (GLIBC_2.2.5) [SUSv4]	<code>wcsftime</code> (GLIBC_2.2.5) [SUSv4]
<code>fprintf</code> (GLIBC_2.2.5) [SUSv4]	<code>posix_spawnattr_init</code> (GLIBC_2.2.5) [SUSv4]	<code>wcslen</code> (GLIBC_2.2.5) [SUSv4]
<code>fputc</code> (GLIBC_2.2.5) [SUSv4]	<code>posix_spawnattr_setflags</code> (GLIBC_2.2.5) [SUSv4]	<code>wcscasecmp</code> (GLIBC_2.2.5) [SUSv4]
<code>fputc_unlocked</code> (GLIBC_2.2.5) [LSB]	<code>posix_spawnattr_setpgroup</code> (GLIBC_2.2.5) [SUSv4]	<code>wcsncat</code> (GLIBC_2.2.5) [SUSv4]
<code>fputs</code> (GLIBC_2.2.5) [SUSv4]	<code>posix_spawnattr_setschedparam</code> (GLIBC_2.2.5) [SUSv4]	<code>wcsncmp</code> (GLIBC_2.2.5) [SUSv4]
<code>fputs_unlocked</code> (GLIBC_2.2.5) [LSB]	<code>posix_spawnattr_setschedpolicy</code> (GLIBC_2.2.5) [SUSv4]	<code>wcsncpy</code> (GLIBC_2.2.5) [SUSv4]
<code>fputwc</code> (GLIBC_2.2.5) [SUSv4]	<code>posix_spawnattr_setsigdefault</code> (GLIBC_2.2.5) [SUSv4]	<code>wcsnlen</code> (GLIBC_2.2.5) [SUSv4]
<code>fputwc_unlocked</code> (GLIBC_2.2.5) [LSB]	<code>posix_spawnattr_setsigmask</code> (GLIBC_2.2.5) [SUSv4]	<code>wcsnrtombs</code> (GLIBC_2.2.5) [SUSv4]
<code>fputws</code> (GLIBC_2.2.5) [SUSv4]	<code>posix_spawnnp</code> (GLIBC_2.15) [SUSv4]	<code>wcsprk</code> (GLIBC_2.2.5) [SUSv4]
<code>fputws_unlocked</code> (GLIBC_2.2.5) [LSB]	<code>pread</code> (GLIBC_2.2.5) [SUSv4]	<code>wcsrchr</code> (GLIBC_2.2.5) [SUSv4]

LSB Core - X86-64 5.0

fread(GLIBC_2.2.5) [SUSv4]	pread64(GLIBC_2.2.5) [LSB]	wcsrtombs(GLIBC_2.2.5) [SUSv4]
fread_unlocked(GLIBC_2.2.5) [LSB]	printf(GLIBC_2.2.5) [SUSv4]	wcsspn(GLIBC_2.2.5) [SUSv4]
free(GLIBC_2.2.5) [SUSv4]	pselect(GLIBC_2.2.5) [SUSv4]	wcsstr(GLIBC_2.2.5) [SUSv4]
freeardrinfo(GLIBC_2.2.5) [SUSv4]	psignal(GLIBC_2.2.5) [SUSv4]	wctod(GLIBC_2.2.5) [SUSv4]
freopen(GLIBC_2.2.5) [SUSv4]	ptrace(GLIBC_2.2.5) [LSB]	wctof(GLIBC_2.2.5) [SUSv4]
freopen64(GLIBC_2.2.5) [LFS]	ptsname(GLIBC_2.2.5) [SUSv4]	wctoimax(GLIBC_2.2.5) [SUSv4]
fscanf(GLIBC_2.2.5) [LSB]	putc(GLIBC_2.2.5) [SUSv4]	wcstok(GLIBC_2.2.5) [SUSv4]
fseek(GLIBC_2.2.5) [SUSv4]	putc_unlocked(GLIBC_2.2.5) [SUSv4]	wctol(GLIBC_2.2.5) [SUSv4]
fseeko(GLIBC_2.2.5) [SUSv4]	putchar(GLIBC_2.2.5) [SUSv4]	wctold(GLIBC_2.2.5) [SUSv4]
fseeko64(GLIBC_2.2.5) [LFS]	putchar_unlocked(GLIBC_2.2.5) [SUSv4]	wctoll(GLIBC_2.2.5) [SUSv4]
fsetpos(GLIBC_2.2.5) [SUSv4]	putenv(GLIBC_2.2.5) [SUSv4]	wcstombs(GLIBC_2.2.5) [SUSv4]
fsetpos64(GLIBC_2.2.5) [LFS]	puts(GLIBC_2.2.5) [SUSv4]	wcstoq(GLIBC_2.2.5) [LSB]
fstatfs(GLIBC_2.2.5) [LSB]	pututxline(GLIBC_2.2.5) [SUSv4]	wctoul(GLIBC_2.2.5) [SUSv4]
fstatfs64(GLIBC_2.2.5) [LSB]	putw(GLIBC_2.2.5) [SUSv2]	wctoull(GLIBC_2.2.5) [SUSv4]
fstatvfs(GLIBC_2.2.5) [SUSv4]	putwc(GLIBC_2.2.5) [SUSv4]	wcstoumax(GLIBC_2.2.5) [SUSv4]
fstatvfs64(GLIBC_2.2.5) [LFS]	putwc_unlocked(GLIBC_2.2.5) [LSB]	wctouq(GLIBC_2.2.5) [LSB]
fsync(GLIBC_2.2.5) [SUSv4]	putwchar(GLIBC_2.2.5) [SUSv4]	wcswcs(GLIBC_2.2.5) [SUSv3]
ftell(GLIBC_2.2.5) [SUSv4]	putwchar_unlocked(GLIBC_2.2.5) [LSB]	wcswidth(GLIBC_2.2.5) [SUSv4]
ftello(GLIBC_2.2.5) [SUSv4]	pwrite(GLIBC_2.2.5) [SUSv4]	wcsxfrm(GLIBC_2.2.5) [SUSv4]
ftello64(GLIBC_2.2.5) [LFS]	pwrite64(GLIBC_2.2.5) [LSB]	wctob(GLIBC_2.2.5) [SUSv4]
ftime(GLIBC_2.2.5) [SUSv3]	qsort(GLIBC_2.2.5) [SUSv4]	wctomb(GLIBC_2.2.5) [SUSv4]
ftok(GLIBC_2.2.5) [SUSv4]	raise(GLIBC_2.2.5) [SUSv4]	wctrans(GLIBC_2.2.5) [SUSv4]
ftruncate(GLIBC_2.2.5) [SUSv4]	rand(GLIBC_2.2.5) [SUSv4]	wctype(GLIBC_2.2.5) [SUSv4]
ftruncate64(GLIBC_2.2.5) [LFS]	rand_r(GLIBC_2.2.5) [SUSv4]	wcwidth(GLIBC_2.2.5) [SUSv4]
ftrylockfile(GLIBC_2.2.5) [SUSv4]	random(GLIBC_2.2.5) [SUSv4]	wmemchr(GLIBC_2.2.5) [SUSv4]

ftw(GLIBC_2.2.5) [SUSv4]	random_r(GLIBC_2.2.5) [LSB]	wmemcmp(GLIBC_2.2.5) [SUSv4]
ftw64(GLIBC_2.2.5) [LFS]	read(GLIBC_2.2.5) [SUSv4]	wmemcpy(GLIBC_2.2.5) [SUSv4]
funlockfile(GLIBC_2.2.5) [SUSv4]	readdir(GLIBC_2.2.5) [SUSv4]	wmemmove(GLIBC_2.2.5) [SUSv4]
fwide(GLIBC_2.2.5) [SUSv4]	readdir64(GLIBC_2.2.5) [LFS]	wmemset(GLIBC_2.2.5) [SUSv4]
fwprintf(GLIBC_2.2.5) [SUSv4]	readdir64_r(GLIBC_2.2.5) [LSB]	wordexp(GLIBC_2.2.5) [SUSv4]
fwrite(GLIBC_2.2.5) [SUSv4]	readdir_r(GLIBC_2.2.5) [SUSv4]	wordfree(GLIBC_2.2.5) [SUSv4]
fwrite_unlocked(GLIBC_2.2.5) [LSB]	readlink(GLIBC_2.2.5) [SUSv4]	wprintf(GLIBC_2.2.5) [SUSv4]
fwscanf(GLIBC_2.2.5) [LSB]	readv(GLIBC_2.2.5) [SUSv4]	write(GLIBC_2.2.5) [SUSv4]
gai_strerror(GLIBC_2.2.5) [SUSv4]	realloc(GLIBC_2.2.5) [SUSv4]	writev(GLIBC_2.2.5) [SUSv4]
gcvt(GLIBC_2.2.5) [SUSv3]	realpath(GLIBC_2.3) [SUSv4]	wscanf(GLIBC_2.2.5) [LSB]
getaddrinfo(GLIBC_2.2.5) [SUSv4]	recv(GLIBC_2.2.5) [SUSv4]	xdr_accepted_reply(GLIBC_2.2.5) [SVID.4]
getc(GLIBC_2.2.5) [SUSv4]	recvfrom(GLIBC_2.2.5) [SUSv4]	xdr_array(GLIBC_2.2.5) [SVID.4]
getc_unlocked(GLIBC_2.2.5) [SUSv4]	recvmsg(GLIBC_2.2.5) [SUSv4]	xdr_bool(GLIBC_2.2.5) [SVID.4]
getchar(GLIBC_2.2.5) [SUSv4]	regcomp(GLIBC_2.2.5) [SUSv4]	xdr_bytes(GLIBC_2.2.5) [SVID.4]
getchar_unlocked(GLIBC_2.2.5) [SUSv4]	regerror(GLIBC_2.2.5) [SUSv4]	xdr_callhdr(GLIBC_2.2.5) [SVID.4]
getcontext(GLIBC_2.2.5) [SUSv3]	regex(GLIBC_2.3.4) [LSB]	xdr_callmsg(GLIBC_2.2.5) [SVID.4]
getcwd(GLIBC_2.2.5) [LSB]	regfree(GLIBC_2.2.5) [SUSv4]	xdr_char(GLIBC_2.2.5) [SVID.4]
getdate(GLIBC_2.2.5) [SUSv4]	remove(GLIBC_2.2.5) [SUSv4]	xdr_double(GLIBC_2.2.5) [SVID.4]
getdelim(GLIBC_2.2.5) [SUSv4]	remque(GLIBC_2.2.5) [SUSv4]	xdr_enum(GLIBC_2.2.5) [SVID.4]
getdomainname(GLIBC_2.2.5) [LSB]	rename(GLIBC_2.2.5) [SUSv4]	xdr_float(GLIBC_2.2.5) [SVID.4]
getdtablesize(GLIBC_2.2.5) [LSB]	rewind(GLIBC_2.2.5) [SUSv4]	xdr_free(GLIBC_2.2.5) [SVID.4]
getegid(GLIBC_2.2.5) [SUSv4]	rewinddir(GLIBC_2.2.5) [SUSv4]	xdr_int(GLIBC_2.2.5) [SVID.4]
getenv(GLIBC_2.2.5) [SUSv4]	rindex(GLIBC_2.2.5) [SUSv3]	xdr_long(GLIBC_2.2.5) [SVID.4]
geteuid(GLIBC_2.2.5) [SUSv4]	rmdir(GLIBC_2.2.5) [SUSv4]	xdr_opaque(GLIBC_2.2.5) [SVID.4]
getgid(GLIBC_2.2.5) [SUSv4]	sbrk(GLIBC_2.2.5) [SUSv2]	xdr_opaque_auth(GLIBC_2.2.5) [SVID.4]

LSB Core - X86-64 5.0

getgrent(GLIBC_2.2.5) [SUSv4]	scandir(GLIBC_2.2.5) [SUSv4]	xdr_pointer(GLIBC_2.2.5) [SVID.4]
getgrent_r(GLIBC_2.2.5) [LSB]	scandir64(GLIBC_2.2.5) [LSB]	xdr_reference(GLIBC_2.2.5) [SVID.4]
getgrgid(GLIBC_2.2.5) [SUSv4]	scanf(GLIBC_2.2.5) [LSB]	xdr_rejected_reply(GLIBC_2.2.5) [SVID.4]
getgrgid_r(GLIBC_2.2.5) [SUSv4]	sched_get_priority_max(GLIBC_2.2.5) [SUSv4]	xdr_replymsg(GLIBC_2.2.5) [SVID.4]
getgrnam(GLIBC_2.2.5) [SUSv4]	sched_get_priority_min(GLIBC_2.2.5) [SUSv4]	xdr_short(GLIBC_2.2.5) [SVID.4]
getgrnam_r(GLIBC_2.2.5) [SUSv4]	sched_getparam(GLIBC_2.2.5) [SUSv4]	xdr_string(GLIBC_2.2.5) [SVID.4]
getgrouplist(GLIBC_2.2.5) [LSB]	sched_getscheduler(GLIBC_2.2.5) [SUSv4]	xdr_u_char(GLIBC_2.2.5) [SVID.4]
getgroups(GLIBC_2.2.5) [SUSv4]	sched_rr_get_interval(GLIBC_2.2.5) [SUSv4]	xdr_u_int(GLIBC_2.2.5) [LSB]
gethostbyaddr(GLIBC_2.2.5) [SUSv3]	sched_setparam(GLIBC_2.2.5) [SUSv4]	xdr_u_long(GLIBC_2.2.5) [SVID.4]
gethostbyaddr_r(GLIBC_2.2.5) [LSB]	sched_setscheduler(GLIBC_2.2.5) [LSB]	xdr_u_short(GLIBC_2.2.5) [SVID.4]
gethostbyname(GLIBC_2.2.5) [SUSv3]	sched_yield(GLIBC_2.2.5) [SUSv4]	xdr_union(GLIBC_2.2.5) [SVID.4]
gethostbyname2(GLIBC_2.2.5) [LSB]	seed48(GLIBC_2.2.5) [SUSv4]	xdr_vector(GLIBC_2.2.5) [SVID.4]
gethostbyname2_r(GLIBC_2.2.5) [LSB]	seed48_r(GLIBC_2.2.5) [LSB]	xdr_void(GLIBC_2.2.5) [SVID.4]
gethostbyname_r(GLIBC_2.2.5) [LSB]	seekdir(GLIBC_2.2.5) [SUSv4]	xdr_wrapstring(GLIBC_2.2.5) [SVID.4]
gethostid(GLIBC_2.2.5) [SUSv4]	select(GLIBC_2.2.5) [SUSv4]	xdrmem_create(GLIBC_2.2.5) [SVID.4]
gethostname(GLIBC_2.2.5) [SUSv4]	semctl(GLIBC_2.2.5) [SUSv4]	xdrrec_create(GLIBC_2.2.5) [SVID.4]
getitimer(GLIBC_2.2.5) [SUSv4]	semget(GLIBC_2.2.5) [SUSv4]	xdrrec_endofrecord(GLIBC_2.2.5) [RPC + XDR]
getline(GLIBC_2.2.5) [SUSv4]	semop(GLIBC_2.2.5) [SUSv4]	xdrrec_eof(GLIBC_2.2.5) [SVID.4]
getloadavg(GLIBC_2.2.5) [LSB]	send(GLIBC_2.2.5) [SUSv4]	xdrrec_skiprecord(GLIBC_2.2.5) [RPC + XDR]
getlogin(GLIBC_2.2.5) [SUSv4]	sendfile(GLIBC_2.2.5) [LSB]	xdrstdio_create(GLIBC_2.2.5) [LSB]
getlogin_r(GLIBC_2.2.5) [SUSv4]	sendmsg(GLIBC_2.2.5) [SUSv4]	
getnameinfo(GLIBC_2.2.5) [SUSv4]	sendto(GLIBC_2.2.5) [SUSv4]	

Table A-2 libc Data Interfaces

__daylight [LSB]	__tzname [LSB]	in6addr_loopback [SUSv3] 1
__environ [LSB]	__sys_errlist [LSB]	
__timezone [LSB]	in6addr_any [SUSv3]	

A.2 libcrypt

The behavior of the interfaces in this library is specified by the following Standards.

[LSB Core - Generic](#) [LSB]

[POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#) [SUSv4]

Table A-3 libcrypt Function Interfaces

crypt(GLIBC_2.2.5) [SUSv4]	encrypt(GLIBC_2.2.5) [SUSv4]	setkey(GLIBC_2.2.5) [SUSv4]
crypt_r(GLIBC_2.2.5) [LSB]	encrypt_r(GLIBC_2.2.5) [LSB]	setkey_r(GLIBC_2.2.5) [LSB]

A.3 libdl

The behavior of the interfaces in this library is specified by the following Standards.

[LSB Core - Generic](#) [LSB]

[POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#) [SUSv4]

Table A-4 libdl Function Interfaces

dldaddr(GLIBC_2.2.5) [LSB]	dlderror(GLIBC_2.2.5) [SUSv4]	dldsym(GLIBC_2.2.5) [LSB]
dldclose(GLIBC_2.2.5) [SUSv4]	dldopen(GLIBC_2.2.5) [LSB]	dldvsym(GLIBC_2.2.5) [LSB]

A.4 libgcc_s

The behavior of the interfaces in this library is specified by the following Standards.

[LSB Core - Generic](#) [LSB]

Table A-5 libgcc_s Function Interfaces

_Unwind_Backtrace(GC C_3.3) [LSB]	_Unwind_GetDataRelBas e(GCC_3.0) [LSB]	_Unwind_RaiseException (GCC_3.0) [LSB]
_Unwind_DeleteExceptio n(GCC_3.0) [LSB]	_Unwind_GetGR(GCC_3 .0) [LSB]	_Unwind_Resume(GCC_ 3.0) [LSB]
_Unwind_FindEnclosing Function(GCC_3.3) [LSB]	_Unwind_GetIP(GCC_3. 0) [LSB]	_Unwind_Resume_or_Re throw(GCC_3.3) [LSB]
_Unwind_Find_FDE(GC C_3.0) [LSB]	_Unwind_GetLanguageS pecificData(GCC_3.0) [LSB]	_Unwind_SetGR(GCC_3. 0) [LSB]
_Unwind_ForcedUnwind(GCC_3.0) [LSB]	_Unwind_GetRegionStart (GCC_3.0) [LSB]	_Unwind_SetIP(GCC_3.0) [LSB]
_Unwind_GetCFA(GCC_ 3.3) [LSB]	_Unwind_GetTextRelBas e(GCC_3.0) [LSB]	

A.5 libm

The behavior of the interfaces in this library is specified by the following Standards.

[LSB Core - Generic](#) [LSB]

[POSIX 1003.1-2001 \(ISO/IEC 9945-2003\)](#) [SUSv3]

[POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#) [SUSv4]

Table A-6 libm Function Interfaces

__finite(GLIBC_2.2.5) [LSB]	csinhl(GLIBC_2.2.5) [SUSv4]	llround(GLIBC_2.2.5) [SUSv4]
__finitef(GLIBC_2.2.5)	csinl(GLIBC_2.2.5)	llroundf(GLIBC_2.2.5)

LSB Core - X86-64 5.0

[LSB]	[SUSv4]	[SUSv4]
__finitel(GLIBC_2.2.5) [LSB]	csqrt(GLIBC_2.2.5) [SUSv4]	llroundl(GLIBC_2.2.5) [SUSv4]
__fpclassify(GLIBC_2.2.5) [LSB]	csqrtf(GLIBC_2.2.5) [SUSv4]	log(GLIBC_2.2.5) [SUSv4]
__fpclassifyf(GLIBC_2.2.5) [LSB]	csqrtl(GLIBC_2.2.5) [SUSv4]	log10(GLIBC_2.2.5) [SUSv4]
__fpclassifyl(GLIBC_2.2.5) [LSB]	ctan(GLIBC_2.2.5) [SUSv4]	log10f(GLIBC_2.2.5) [SUSv4]
__signbit(GLIBC_2.2.5) [LSB]	ctanf(GLIBC_2.2.5) [SUSv4]	log10l(GLIBC_2.2.5) [SUSv4]
__signbitf(GLIBC_2.2.5) [LSB]	ctanh(GLIBC_2.2.5) [SUSv4]	log1p(GLIBC_2.2.5) [SUSv4]
__signbitl(GLIBC_2.2.5) [LSB]	ctanhf(GLIBC_2.2.5) [SUSv4]	log1pf(GLIBC_2.2.5) [SUSv4]
acos(GLIBC_2.2.5) [SUSv4]	ctanhl(GLIBC_2.2.5) [SUSv4]	log1pl(GLIBC_2.2.5) [SUSv4]
acosf(GLIBC_2.2.5) [SUSv4]	ctanl(GLIBC_2.2.5) [SUSv4]	log2(GLIBC_2.2.5) [SUSv4]
acosh(GLIBC_2.2.5) [SUSv4]	drem(GLIBC_2.2.5) [LSB]	log2f(GLIBC_2.2.5) [SUSv4]
acoshf(GLIBC_2.2.5) [SUSv4]	dremf(GLIBC_2.2.5) [LSB]	log2l(GLIBC_2.2.5) [SUSv4]
acoshl(GLIBC_2.2.5) [SUSv4]	dreml(GLIBC_2.2.5) [LSB]	logb(GLIBC_2.2.5) [SUSv4]
acosl(GLIBC_2.2.5) [SUSv4]	erf(GLIBC_2.2.5) [SUSv4]	logbf(GLIBC_2.2.5) [SUSv4]
asin(GLIBC_2.2.5) [SUSv4]	erfc(GLIBC_2.2.5) [SUSv4]	logbl(GLIBC_2.2.5) [SUSv4]
asinf(GLIBC_2.2.5) [SUSv4]	erfcf(GLIBC_2.2.5) [SUSv4]	logf(GLIBC_2.2.5) [SUSv4]
asinh(GLIBC_2.2.5) [SUSv4]	erfcl(GLIBC_2.2.5) [SUSv4]	logl(GLIBC_2.2.5) [SUSv4]
asinhf(GLIBC_2.2.5) [SUSv4]	erff(GLIBC_2.2.5) [SUSv4]	lrint(GLIBC_2.2.5) [SUSv4]
asinhl(GLIBC_2.2.5) [SUSv4]	erfl(GLIBC_2.2.5) [SUSv4]	lrintf(GLIBC_2.2.5) [SUSv4]
asinl(GLIBC_2.2.5) [SUSv4]	exp(GLIBC_2.2.5) [SUSv4]	lrintl(GLIBC_2.2.5) [SUSv4]
atan(GLIBC_2.2.5) [SUSv4]	exp10(GLIBC_2.2.5) [LSB]	lround(GLIBC_2.2.5) [SUSv4]
atan2(GLIBC_2.2.5) [SUSv4]	exp10f(GLIBC_2.2.5) [LSB]	lroundf(GLIBC_2.2.5) [SUSv4]
atan2f(GLIBC_2.2.5) [SUSv4]	exp10l(GLIBC_2.2.5) [LSB]	lroundl(GLIBC_2.2.5) [SUSv4]
atan2l(GLIBC_2.2.5) [SUSv4]	exp2(GLIBC_2.2.5) [SUSv4]	matherr(GLIBC_2.2.5) [LSB]
atanf(GLIBC_2.2.5) [SUSv4]	exp2f(GLIBC_2.2.5) [SUSv4]	modf(GLIBC_2.2.5) [SUSv4]

atanh(GLIBC_2.2.5) [SUSv4]	exp2l(GLIBC_2.2.5) [SUSv4]	modff(GLIBC_2.2.5) [SUSv4]
atanhf(GLIBC_2.2.5) [SUSv4]	expf(GLIBC_2.2.5) [SUSv4]	modfl(GLIBC_2.2.5) [SUSv4]
atanhl(GLIBC_2.2.5) [SUSv4]	expl(GLIBC_2.2.5) [SUSv4]	nan(GLIBC_2.2.5) [SUSv4]
atanl(GLIBC_2.2.5) [SUSv4]	expm1(GLIBC_2.2.5) [SUSv4]	nanf(GLIBC_2.2.5) [SUSv4]
cabs(GLIBC_2.2.5) [SUSv4]	expm1f(GLIBC_2.2.5) [SUSv4]	nanl(GLIBC_2.2.5) [SUSv4]
cabsf(GLIBC_2.2.5) [SUSv4]	expm1l(GLIBC_2.2.5) [SUSv4]	nearbyint(GLIBC_2.2.5) [SUSv4]
cabsl(GLIBC_2.2.5) [SUSv4]	fabs(GLIBC_2.2.5) [SUSv4]	nearbyintf(GLIBC_2.2.5) [SUSv4]
cacos(GLIBC_2.2.5) [SUSv4]	fabsf(GLIBC_2.2.5) [SUSv4]	nearbyintl(GLIBC_2.2.5) [SUSv4]
cacosh(GLIBC_2.2.5) [SUSv4]	fabsl(GLIBC_2.2.5) [SUSv4]	nextafter(GLIBC_2.2.5) [SUSv4]
cacoshf(GLIBC_2.2.5) [SUSv4]	fdim(GLIBC_2.2.5) [SUSv4]	nextafterf(GLIBC_2.2.5) [SUSv4]
cacoshl(GLIBC_2.2.5) [SUSv4]	fdimf(GLIBC_2.2.5) [SUSv4]	nextafterl(GLIBC_2.2.5) [SUSv4]
cacosl(GLIBC_2.2.5) [SUSv4]	fdiml(GLIBC_2.2.5) [SUSv4]	nexttoward(GLIBC_2.2.5) [SUSv4]
carg(GLIBC_2.2.5) [SUSv4]	feclearexcept(GLIBC_2.2.5) [SUSv4]	nexttowardf(GLIBC_2.2.5) [SUSv4]
cargf(GLIBC_2.2.5) [SUSv4]	fedisableexcept(GLIBC_2.2.5) [LSB]	nexttowardl(GLIBC_2.2.5) [SUSv4]
cargl(GLIBC_2.2.5) [SUSv4]	feenableexcept(GLIBC_2.2.5) [LSB]	pow(GLIBC_2.2.5) [SUSv4]
casin(GLIBC_2.2.5) [SUSv4]	fegetenv(GLIBC_2.2.5) [SUSv4]	pow10(GLIBC_2.2.5) [LSB]
casinf(GLIBC_2.2.5) [SUSv4]	fegetexcept(GLIBC_2.2.5) [LSB]	pow10f(GLIBC_2.2.5) [LSB]
casinh(GLIBC_2.2.5) [SUSv4]	fegetexceptflag(GLIBC_2.2.5) [SUSv4]	pow10l(GLIBC_2.2.5) [LSB]
casinhf(GLIBC_2.2.5) [SUSv4]	fegetround(GLIBC_2.2.5) [SUSv4]	powf(GLIBC_2.2.5) [SUSv4]
casinhl(GLIBC_2.2.5) [SUSv4]	feholdexcept(GLIBC_2.2.5) [SUSv4]	powl(GLIBC_2.2.5) [SUSv4]
casinl(GLIBC_2.2.5) [SUSv4]	feraiseexcept(GLIBC_2.2.5) [SUSv4]	remainder(GLIBC_2.2.5) [SUSv4]
catan(GLIBC_2.2.5) [SUSv4]	fesetenv(GLIBC_2.2.5) [SUSv4]	remainderf(GLIBC_2.2.5) [SUSv4]
catanf(GLIBC_2.2.5) [SUSv4]	fesetexceptflag(GLIBC_2.2.5) [SUSv4]	remainderl(GLIBC_2.2.5) [SUSv4]
catanh(GLIBC_2.2.5) [SUSv4]	fesetround(GLIBC_2.2.5) [SUSv4]	remquo(GLIBC_2.2.5) [SUSv4]
	fetestexcept(GLIBC_2.2.5) [SUSv4]	remquof(GLIBC_2.2.5) [SUSv4]

LSB Core - X86-64 5.0

catanhf(GLIBC_2.2.5) [SUSv4]	feupdateenv(GLIBC_2.2.5) [SUSv4]	remquo(GLIBC_2.2.5) [SUSv4]
catanh(GLIBC_2.2.5) [SUSv4]	finite(GLIBC_2.2.5) [LSB]	rint(GLIBC_2.2.5) [SUSv4]
catanl(GLIBC_2.2.5) [SUSv4]	finitef(GLIBC_2.2.5) [LSB]	rintf(GLIBC_2.2.5) [SUSv4]
cbrt(GLIBC_2.2.5) [SUSv4]	finitel(GLIBC_2.2.5) [LSB]	rintl(GLIBC_2.2.5) [SUSv4]
cbrtf(GLIBC_2.2.5) [SUSv4]	floor(GLIBC_2.2.5) [SUSv4]	round(GLIBC_2.2.5) [SUSv4]
cbrtl(GLIBC_2.2.5) [SUSv4]	floorf(GLIBC_2.2.5) [SUSv4]	roundf(GLIBC_2.2.5) [SUSv4]
ccos(GLIBC_2.2.5) [SUSv4]	floorl(GLIBC_2.2.5) [SUSv4]	roundl(GLIBC_2.2.5) [SUSv4]
ccosf(GLIBC_2.2.5) [SUSv4]	fma(GLIBC_2.2.5) [SUSv4]	scalb(GLIBC_2.2.5) [SUSv3]
ccosh(GLIBC_2.2.5) [SUSv4]	fmaf(GLIBC_2.2.5) [SUSv4]	scalbf(GLIBC_2.2.5) [LSB]
ccoshf(GLIBC_2.2.5) [SUSv4]	fmal(GLIBC_2.2.5) [SUSv4]	scalbl(GLIBC_2.2.5) [LSB]
ccoshl(GLIBC_2.2.5) [SUSv4]	fmax(GLIBC_2.2.5) [SUSv4]	scalbln(GLIBC_2.2.5) [SUSv4]
ccosl(GLIBC_2.2.5) [SUSv4]	fmaxf(GLIBC_2.2.5) [SUSv4]	scalblnf(GLIBC_2.2.5) [SUSv4]
ceil(GLIBC_2.2.5) [SUSv4]	fmaxl(GLIBC_2.2.5) [SUSv4]	scalblnl(GLIBC_2.2.5) [SUSv4]
ceilf(GLIBC_2.2.5) [SUSv4]	fmin(GLIBC_2.2.5) [SUSv4]	scalbn(GLIBC_2.2.5) [SUSv4]
ceill(GLIBC_2.2.5) [SUSv4]	fminf(GLIBC_2.2.5) [SUSv4]	scalbnf(GLIBC_2.2.5) [SUSv4]
cexp(GLIBC_2.2.5) [SUSv4]	fminl(GLIBC_2.2.5) [SUSv4]	scalbnl(GLIBC_2.2.5) [SUSv4]
cexpf(GLIBC_2.2.5) [SUSv4]	fmod(GLIBC_2.2.5) [SUSv4]	significand(GLIBC_2.2.5) [LSB]
cexpl(GLIBC_2.2.5) [SUSv4]	fmodf(GLIBC_2.2.5) [SUSv4]	significandf(GLIBC_2.2.5) [LSB]
cimag(GLIBC_2.2.5) [SUSv4]	fmodl(GLIBC_2.2.5) [SUSv4]	significandl(GLIBC_2.2.5) [LSB]
cimagf(GLIBC_2.2.5) [SUSv4]	frexp(GLIBC_2.2.5) [SUSv4]	sin(GLIBC_2.2.5) [SUSv4]
cimagl(GLIBC_2.2.5) [SUSv4]	frexpf(GLIBC_2.2.5) [SUSv4]	sincos(GLIBC_2.2.5) [LSB]
clog(GLIBC_2.2.5) [SUSv4]	frexpl(GLIBC_2.2.5) [SUSv4]	sincosf(GLIBC_2.2.5) [LSB]
clog10(GLIBC_2.2.5) [LSB]	gamma(GLIBC_2.2.5) [LSB]	sincosl(GLIBC_2.2.5) [LSB]
clog10f(GLIBC_2.2.5) [LSB]	gammaf(GLIBC_2.2.5) [LSB]	sinf(GLIBC_2.2.5) [SUSv4]
clog10l(GLIBC_2.2.5) [LSB]	gammal(GLIBC_2.2.5) [LSB]	sinh(GLIBC_2.2.5) [SUSv4]

clogf(GLIBC_2.2.5) [SUSv4]	hypot(GLIBC_2.2.5) [SUSv4]	sinhf(GLIBC_2.2.5) [SUSv4]
clogl(GLIBC_2.2.5) [SUSv4]	hypotf(GLIBC_2.2.5) [SUSv4]	sinhl(GLIBC_2.2.5) [SUSv4]
conj(GLIBC_2.2.5) [SUSv4]	hypotl(GLIBC_2.2.5) [SUSv4]	sinl(GLIBC_2.2.5) [SUSv4]
conjf(GLIBC_2.2.5) [SUSv4]	ilogb(GLIBC_2.2.5) [SUSv4]	sqrt(GLIBC_2.2.5) [SUSv4]
conjl(GLIBC_2.2.5) [SUSv4]	ilogbf(GLIBC_2.2.5) [SUSv4]	sqrtf(GLIBC_2.2.5) [SUSv4]
copysign(GLIBC_2.2.5) [SUSv4]	ilogbl(GLIBC_2.2.5) [SUSv4]	sqrtl(GLIBC_2.2.5) [SUSv4]
copysignf(GLIBC_2.2.5) [SUSv4]	j0(GLIBC_2.2.5)[SUSv4]	tan(GLIBC_2.2.5) [SUSv4]
copysignl(GLIBC_2.2.5) [SUSv4]	j0f(GLIBC_2.2.5)[LSB]	tanf(GLIBC_2.2.5) [SUSv4]
cos(GLIBC_2.2.5) [SUSv4]	j0l(GLIBC_2.2.5)[LSB]	tanh(GLIBC_2.2.5) [SUSv4]
cosf(GLIBC_2.2.5) [SUSv4]	j1(GLIBC_2.2.5)[SUSv4]	tanhf(GLIBC_2.2.5) [SUSv4]
cosh(GLIBC_2.2.5) [SUSv4]	j1f(GLIBC_2.2.5)[LSB]	tanhl(GLIBC_2.2.5) [SUSv4]
coshf(GLIBC_2.2.5) [SUSv4]	j1l(GLIBC_2.2.5)[LSB]	tanl(GLIBC_2.2.5) [SUSv4]
coshl(GLIBC_2.2.5) [SUSv4]	jnl(GLIBC_2.2.5)[SUSv4]	tgamma(GLIBC_2.2.5) [SUSv4]
cosl(GLIBC_2.2.5) [SUSv4]	jnf(GLIBC_2.2.5)[LSB]	tgammaf(GLIBC_2.2.5) [SUSv4]
cpow(GLIBC_2.2.5) [SUSv4]	jnl(GLIBC_2.2.5)[LSB]	tgamma1(GLIBC_2.2.5) [SUSv4]
cpowf(GLIBC_2.2.5) [SUSv4]	ldexp(GLIBC_2.2.5) [SUSv4]	trunc(GLIBC_2.2.5) [SUSv4]
cpowl(GLIBC_2.2.5) [SUSv4]	ldexpf(GLIBC_2.2.5) [SUSv4]	truncf(GLIBC_2.2.5) [SUSv4]
cproj(GLIBC_2.2.5) [SUSv4]	ldexpl(GLIBC_2.2.5) [SUSv4]	truncl(GLIBC_2.2.5) [SUSv4]
cprojf(GLIBC_2.2.5) [SUSv4]	lgamma(GLIBC_2.2.5) [SUSv4]	y0(GLIBC_2.2.5) [SUSv4]
cprojl(GLIBC_2.2.5) [SUSv4]	lgamma_r(GLIBC_2.2.5) [LSB]	y0f(GLIBC_2.2.5)[LSB]
creal(GLIBC_2.2.5) [SUSv4]	lgammaf(GLIBC_2.2.5) [SUSv4]	y0l(GLIBC_2.2.5)[LSB]
crealf(GLIBC_2.2.5) [SUSv4]	lgammaf_r(GLIBC_2.2.5) [LSB]	y1(GLIBC_2.2.5) [SUSv4]
creall(GLIBC_2.2.5) [SUSv4]	lgamma1(GLIBC_2.2.5) [SUSv4]	y1f(GLIBC_2.2.5)[LSB]
csin(GLIBC_2.2.5) [SUSv4]	lgamma1_r(GLIBC_2.2.5) [LSB]	y1l(GLIBC_2.2.5)[LSB]
csinf(GLIBC_2.2.5) [SUSv4]	llrint(GLIBC_2.2.5) [SUSv4]	yn(GLIBC_2.2.5) [SUSv4]

csinh(GLIBC_2.2.5) [SUSv4]	llrintf(GLIBC_2.2.5) [SUSv4]	ynf(GLIBC_2.2.5) [LSB]
csinhf(GLIBC_2.2.5) [SUSv4]	llrintl(GLIBC_2.2.5) [SUSv4]	ynl(GLIBC_2.2.5) [LSB]

Table A-7 libm Data Interfaces

signgam [SUSv4]		
---	--	--

A.6 libpthread

The behavior of the interfaces in this library is specified by the following Standards.

[Large File Support](#) [LFS]

[LSB Core - Generic](#) [LSB]

[POSIX 1003.1-2001 \(ISO/IEC 9945-2003\)](#) [SUSv3]

[POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#) [SUSv4]

Table A-8 libpthread Function Interfaces

__errno_location(GLIBC_2.2.5) [LSB]	pthread_barrierattr_setps hared(GLIBC_2.2.5) [SUSv4]	pthread_rwlockattr_destro y(GLIBC_2.2.5) [SUSv4]
__h_errno_location(GLIB C_2.2.5) [LSB]	pthread_cancel(GLIBC_2 .2.5) [SUSv4]	pthread_rwlockattr_getki nd_np(GLIBC_2.2.5) [LSB]
__libc_current_sigrtmax(GLIBC_2.2.5) [LSB]	pthread_cond_broadcast(GLIBC_2.3.2) [SUSv4]	pthread_rwlockattr_getps hared(GLIBC_2.2.5) [SUSv4]
__libc_current_sigrtmin(GLIBC_2.2.5) [LSB]	pthread_cond_destroy(GL IBC_2.3.2) [SUSv4]	pthread_rwlockattr_init(G LIBC_2.2.5) [SUSv4]
_pthread_cleanup_pop(G LIBC_2.2.5) [LSB]	pthread_cond_init(GLIB C_2.3.2) [SUSv4]	pthread_rwlockattr_setkin d_np(GLIBC_2.2.5) [LSB]
_pthread_cleanup_push(G LIBC_2.2.5) [LSB]	pthread_cond_signal(GLI BC_2.3.2) [SUSv4]	pthread_rwlockattr_setps hared(GLIBC_2.2.5) [SUSv4]
accept(GLIBC_2.2.5) [SUSv4]	pthread_cond_timedwait(GLIBC_2.3.2) [SUSv4]	pthread_self(GLIBC_2.2. 5) [SUSv4]
close(GLIBC_2.2.5) [SUSv4]	pthread_cond_wait(GLIB C_2.3.2) [SUSv4]	pthread_setcancelstate(G LIBC_2.2.5) [SUSv4]
connect(GLIBC_2.2.5) [SUSv4]	pthread_condattr_destroy(GLIBC_2.2.5) [SUSv4]	pthread_setcanceltype(GL IBC_2.2.5) [SUSv4]
fcntl(GLIBC_2.2.5) [LSB]	pthread_condattr_getps hared(GLIBC_2.2.5) [SUSv4]	pthread_setconcurrency(GLIBC_2.2.5) [SUSv4]
flockfile(GLIBC_2.2.5) [SUSv4]	pthread_condattr_init(GL IBC_2.2.5) [SUSv4]	pthread_setschedparam(G LIBC_2.2.5) [SUSv4]
fork(GLIBC_2.2.5) [SUSv4]	pthread_condattr_setps hared(GLIBC_2.2.5) [SUSv4]	pthread_setspecific(GLIB C_2.2.5) [SUSv4]
fsync(GLIBC_2.2.5) [SUSv4]	pthread_create(GLIBC_2. 2.5) [SUSv4]	pthread_sigmask(GLIBC _2.2.5) [SUSv4]
ftrylockfile(GLIBC_2.2.5) [SUSv4]	pthread_detach(GLIBC_2 .2.5) [SUSv4]	pthread_spin_destroy(GL IBC_2.2.5) [SUSv4]

funlockfile(GLIBC_2.2.5) [SUSv4]	pthread_equal(GLIBC_2.2.5) [SUSv4]	pthread_spin_init(GLIBC_2.2.5) [SUSv4]
longjmp(GLIBC_2.2.5) [SUSv4]	pthread_exit(GLIBC_2.2.5) [SUSv4]	pthread_spin_lock(GLIBC_2.2.5) [SUSv4]
lseek(GLIBC_2.2.5) [SUSv4]	pthread_getattr_np(GLIBC_2.2.5) [LSB]	pthread_spin_trylock(GLIBC_2.2.5) [SUSv4]
lseek64(GLIBC_2.2.5) [LFS]	pthread_getconcurrency(GLIBC_2.2.5) [SUSv4]	pthread_spin_unlock(GLIBC_2.2.5) [SUSv4]
msync(GLIBC_2.2.5) [SUSv4]	pthread_getcpuclockid(GLIBC_2.2.5) [SUSv4]	pthread_testcancel(GLIBC_2.2.5) [SUSv4]
nanosleep(GLIBC_2.2.5) [SUSv4]	pthread_getschedparam(GLIBC_2.2.5) [SUSv4]	pwrite(GLIBC_2.2.5) [SUSv4]
open(GLIBC_2.2.5) [SUSv4]	pthread_getspecific(GLIBC_2.2.5) [SUSv4]	pwrite64(GLIBC_2.2.5) [LSB]
open64(GLIBC_2.2.5) [LFS]	pthread_join(GLIBC_2.2.5) [SUSv4]	raise(GLIBC_2.2.5) [SUSv4]
pause(GLIBC_2.2.5) [SUSv4]	pthread_key_create(GLIBC_2.2.5) [SUSv4]	read(GLIBC_2.2.5) [SUSv4]
pread(GLIBC_2.2.5) [SUSv4]	pthread_key_delete(GLIBC_2.2.5) [SUSv4]	recv(GLIBC_2.2.5) [SUSv4]
pread64(GLIBC_2.2.5) [LSB]	pthread_kill(GLIBC_2.2.5) [SUSv4]	recvfrom(GLIBC_2.2.5) [SUSv4]
pthread_attr_destroy(GLIBC_2.2.5) [SUSv4]	pthread_mutex_consistent_np(GLIBC_2.4) [LSB]	recvmsg(GLIBC_2.2.5) [SUSv4]
pthread_attr_getdetachstate(GLIBC_2.2.5) [SUSv4]	pthread_mutex_destroy(GLIBC_2.2.5) [SUSv4]	sem_close(GLIBC_2.2.5) [SUSv4]
pthread_attr_getguardsize(GLIBC_2.2.5) [SUSv4]	pthread_mutex_init(GLIBC_2.2.5) [SUSv4]	sem_destroy(GLIBC_2.2.5) [SUSv4]
pthread_attr_getinheritsched(GLIBC_2.2.5) [SUSv4]	pthread_mutex_lock(GLIBC_2.2.5) [SUSv4]	sem_getvalue(GLIBC_2.2.5) [SUSv4]
pthread_attr_getschedparam(GLIBC_2.2.5) [SUSv4]	pthread_mutex_timedlock(GLIBC_2.2.5) [SUSv4]	sem_init(GLIBC_2.2.5) [SUSv4]
pthread_attr_getschedpolicy(GLIBC_2.2.5) [SUSv4]	pthread_mutex_trylock(GLIBC_2.2.5) [SUSv4]	sem_open(GLIBC_2.2.5) [SUSv4]
pthread_attr_getscope(GLIBC_2.2.5) [SUSv4]	pthread_mutex_unlock(GLIBC_2.2.5) [SUSv4]	sem_post(GLIBC_2.2.5) [SUSv4]
pthread_attr_getstack(GLIBC_2.2.5) [SUSv4]	pthread_mutexattr_destroy(GLIBC_2.2.5) [SUSv4]	sem_timedwait(GLIBC_2.2.5) [SUSv4]
pthread_attr_getstackaddr(GLIBC_2.2.5) [SUSv3]	pthread_mutexattr_getpshared(GLIBC_2.2.5) [SUSv4]	sem_trywait(GLIBC_2.2.5) [SUSv4]
pthread_attr_getstacksize(GLIBC_2.2.5) [SUSv4]	pthread_mutexattr_getrobust_np(GLIBC_2.4) [LSB]	sem_unlink(GLIBC_2.2.5) [SUSv4]
pthread_attr_init(GLIBC_2.2.5) [SUSv4]	pthread_mutexattr_gettype(GLIBC_2.2.5) [SUSv4]	sem_wait(GLIBC_2.2.5) [SUSv4]
pthread_attr_setdetachstate(GLIBC_2.2.5) [SUSv4]	pthread_mutexattr_init(GLIBC_2.2.5) [SUSv4]	send(GLIBC_2.2.5) [SUSv4]

pthread_attr_setguardsize(GLIBC_2.2.5)[SUSv4]	pthread_mutexattr_setpsh ared(GLIBC_2.2.5) [SUSv4]	sendmsg(GLIBC_2.2.5) [SUSv4]
pthread_attr_setinheritsched(GLIBC_2.2.5) [SUSv4]	pthread_mutexattr_setrobust_np(GLIBC_2.4) [LSB]	sendto(GLIBC_2.2.5) [SUSv4]
pthread_attr_setschedparam(GLIBC_2.2.5)[SUSv4]	pthread_mutexattr_settype(GLIBC_2.2.5)[SUSv4]	sigaction(GLIBC_2.2.5) [SUSv4]
pthread_attr_setschedpolicy(GLIBC_2.2.5) [SUSv4]	pthread_once(GLIBC_2.2.5) [SUSv4]	siglongjmp(GLIBC_2.2.5) [SUSv4]
pthread_attr_setscope(GLIBC_2.2.5)[SUSv4]	pthread_rwlock_destroy(GLIBC_2.2.5)[SUSv4]	sigwait(GLIBC_2.2.5) [SUSv4]
pthread_attr_setstack(GLIBC_2.2.5)[SUSv4]	pthread_rwlock_init(GLIBC_2.2.5)[SUSv4]	system(GLIBC_2.2.5) [LSB]
pthread_attr_setstackaddr(GLIBC_2.2.5)[SUSv3]	pthread_rwlock_rdlock(GLIBC_2.2.5)[SUSv4]	tcdrain(GLIBC_2.2.5) [SUSv4]
pthread_attr_setstacksize(GLIBC_2.2.5)[SUSv4]	pthread_rwlock_timedrdlock(GLIBC_2.2.5) [SUSv4]	vfork(GLIBC_2.2.5) [SUSv3]
pthread_barrier_destroy(GLIBC_2.2.5)[SUSv4]	pthread_rwlock_timedwrl ock(GLIBC_2.2.5) [SUSv4]	wait(GLIBC_2.2.5) [SUSv4]
pthread_barrier_init(GLIBC_2.2.5)[SUSv4]	pthread_rwlock_tryrdlock(GLIBC_2.2.5)[SUSv4]	waitpid(GLIBC_2.2.5) [LSB]
pthread_barrier_wait(GLIBC_2.2.5)[SUSv4]	pthread_rwlock_trywrlock(GLIBC_2.2.5)[SUSv4]	write(GLIBC_2.2.5) [SUSv4]
pthread_barrierattr_destroy(GLIBC_2.2.5)[SUSv4]	pthread_rwlock_unlock(GLIBC_2.2.5)[SUSv4]	
pthread_barrierattr_init(GLIBC_2.2.5)[SUSv4]	pthread_rwlock_wrlock(GLIBC_2.2.5)[SUSv4]	

A.7 librt

The behavior of the interfaces in this library is specified by the following Standards.

[Large File Support](#) [LFS]

[POSIX 1003.1-2008 \(ISO/IEC 9945-2009\)](#) [SUSv4]

Table A-9 librt Function Interfaces

aio_cancel(GLIBC_2.2.5) [SUSv4]	aio_return64(GLIBC_2.2.5) [LFS]	clock_settime(GLIBC_2.2.5) [SUSv4]
aio_cancel64(GLIBC_2.2.5) [LFS]	aio_suspend(GLIBC_2.2.5) [SUSv4]	shm_open(GLIBC_2.2.5) [SUSv4]
aio_error(GLIBC_2.2.5) [SUSv4]	aio_suspend64(GLIBC_2.2.5) [LFS]	shm_unlink(GLIBC_2.2.5) [SUSv4]
aio_error64(GLIBC_2.2.5) [LFS]	aio_write(GLIBC_2.2.5) [SUSv4]	timer_create(GLIBC_2.3.3) [SUSv4]
aio_fsync(GLIBC_2.2.5) [SUSv4]	aio_write64(GLIBC_2.2.5) [LFS]	timer_delete(GLIBC_2.3.3) [SUSv4]
aio_fsync64(GLIBC_2.2.5) [LFS]	clock_getcpuclockid(GLIBC_2.2.5) [SUSv4]	timer_getoverrun(GLIBC_2.3.3) [SUSv4]
aio_read(GLIBC_2.2.5)	clock_getres(GLIBC_2.2.5)	timer_gettime(GLIBC_2.2.5)

[SUSv4]	5) [SUSv4]	3.3) [SUSv4]
aio_read64(GLIBC_2.2.5) [LFS]	clock_gettime(GLIBC_2.2.5) [SUSv4]	timer_settime(GLIBC_2.3.3) [SUSv4]
aio_return(GLIBC_2.2.5) [SUSv4]	clock_nanosleep(GLIBC_2.2.5) [SUSv4]	

A.8 libutil

The behavior of the interfaces in this library is specified by the following Standards.
[LSB Core - Generic](#) [\[LSB\]](#)

Table A-10 libutil Function Interfaces

forkpty(GLIBC_2.2.5) [LSB]	login_tty(GLIBC_2.2.5) [LSB]	logwtmp(GLIBC_2.2.5) [LSB]
login(GLIBC_2.2.5) [LSB]	logout(GLIBC_2.2.5) [LSB]	openpty(GLIBC_2.2.5) [LSB]

Annex B GNU Free Documentation License (Informative)

This specification is published under the terms of the GNU Free Documentation License, Version 1.1, March 2000

Copyright (C) 2000 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

B.1 PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

B.2 APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent"

is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

B.3 VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

B.4 COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

B.5 MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These

titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

B.6 COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

B.7 COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

B.8 AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

B.9 TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

B.10 TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

B.11 FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

B.12 How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.