LSB Core - Generic 5.0

14.16 Data Definitions for libdl .................................................................
14.17 Interface Definitions for libdl ...........................................................
14.18 Interfaces for librt ............................................................................
14.19 Data Definitions for librt .................................................................
14.20 Interfaces for libcrypt ......................................................................
14.21 Data Definitions for libcrypt .............................................................
14.22 Interface Definitions for libcrypt .......................................................  
14.23 Interfaces for libpam .........................................................................
14.24 Data Definitions for libpam ...............................................................  
14.25 Interface Definitions for libpam ........................................................

IV Utility Libraries .................................................................................

15 Utility Libraries ................................................................................
15.1 Introduction ......................................................................................
15.2 Interfaces for libc ............................................................................
15.3 Data Definitions for libc ..................................................................
15.4 Interface Definitions for libc ...........................................................
15.5 Interfaces for libncurses .................................................................
15.6 Data Definitions for libncurses .........................................................
15.7 Interface Definitions for libncurses ................................................
15.8 Interfaces for libncursesx ...............................................................  
15.9 Data Definitions for libncursesx ........................................................
15.10 Interface Definitions for libncursesx .................................................
15.11 Interfaces for libutil ........................................................................
15.12 Data Definitions for libutil .............................................................
15.13 Interface Definitions for libutil ........................................................

V C++ Libraries ....................................................................................

16 Libraries ............................................................................................
16.1 Interfaces for libstdcxx .................................................................
16.2 Interface Definitions for libstdcxx ...................................................

VI Commands and Utilities .................................................................

17 Commands and Utilities ....................................................................
17.1 Commands and Utilities .................................................................
17.2 Command Behavior ........................................................................

VII Execution Environment ...............................................................

18 File System Hierarchy ......................................................................
18.1 /dev: Device Files ...........................................................................
18.2 /etc: Host-specific system configuration ...........................................
18.3 User Accounting Databases .............................................................
18.4 Path For System Administration Utilities ........................................

19 Additional Recommendations ..........................................................
19.1 Recommendations for applications on ownership and permissions ....

20 Additional Behaviors .........................................................................
20.1 Mandatory Optional Behaviors .......................................................  
20.2 Optional Mandatory Behaviors .......................................................  
20.3 Executable Scripts ..........................................................................  

21 Localization ......................................................................................
21.1 Introduction ......................................................................................
21.2 Regular Expressions ........................................................................
21.3 Pattern Matching Notation .............................................................

VIII System Initialization .......................................................................  

22 System Initialization ..........................................................................  
22.1 Cron Jobs ........................................................................................
22.2 Init Script Actions ..........................................................................  
22.3 Comment Conventions for Init Scripts ..........................................  
22.4 Installation and Removal of Init Scripts .........................................
22.5 Run Levels ......................................................................................
List of Figures

10-1 Version Definition Entries
10-2 Version Definition Auxiliary Entries
10-3 Version Needed Entries
10-4 Version Needed Auxiliary Entries
11-1 Dynamic Structure
12-1 Category 1 Virtual Table
12-2 Category 2 Virtual Table
12-3 Run-Time Type Information Prefix
12-4 Run-Time Type Information For Classes with no base class
12-5 Run-Time Type Information for Classes with a single base class
12-6 Run-Time Type Information for classes with multiple inheritance
12-7 Run-Time Type Information for pointer types
12-8 Run-Time Type Information for pointer to member types
Foreword

This is version 5.0 of the Linux Standard Base Core Specification, Generic Part. 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 common part of the Core module of the Linux Standard Base (LSB), LSB Core - Generic. This module provides the fundamental system interfaces, libraries, and runtime environment upon which all conforming applications and libraries depend.

LSB Core - Generic, the common part, should be used in conjunction with an architecture-specific 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. Architecture-specific parts of the LSB Core Specification 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>
<thead>
<tr>
<th>Name</th>
<th>Title</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filesystem Hierarchy Standard</td>
<td>Filesystem Hierarchy Standard (FHS) 3.0</td>
<td><a href="http://refspecs.linuxbase.org/fhs">http://refspecs.linuxbase.org/fhs</a></td>
</tr>
<tr>
<td>Itanium™ C++ ABI</td>
<td>Itanium™ C++ ABI (Revision 1.86)</td>
<td><a href="http://refspecs.linuxfoundation.org/cxxabi-1.86.html">http://refspecs.linuxfoundation.org/cxxabi-1.86.html</a></td>
</tr>
<tr>
<td>Libncursesw API</td>
<td>Libncursesw API</td>
<td><a href="http://invisible-island.net/ncurses/man/ncurses.3x.html">http://invisible-island.net/ncurses/man/ncurses.3x.html</a></td>
</tr>
<tr>
<td></td>
<td>ISO/IEC 9945-4:2003 Information technology --</td>
<td></td>
</tr>
</tbody>
</table>
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>
<thead>
<tr>
<th>Name</th>
<th>Title</th>
<th>URL</th>
</tr>
</thead>
</table>

© 2015 Linux Foundation
<p>| Linux Assigned Names And Numbers Authority | Linux Assigned Names And Numbers Authority | <a href="http://www.lanana.org/">http://www.lanana.org/</a> |
| Mozilla's NSS SSL Reference | Mozilla's NSS SSL Reference | <a href="http://www.mozilla.org/projects/security/pki/nss/ref/ssl/">http://www.mozilla.org/projects/security/pki/nss/ref/ssl/</a> |
| PAM | Open Software Foundation, Request For Comments: 86.0, October 1995, V. Samar &amp; R. Schemers (SunSoft) | <a href="http://www.opengroup.org/tech/rfc/mirror-rfc/rfc86.0.txt">http://www.opengroup.org/tech/rfc/mirror-rfc/rfc86.0.txt</a> |</p>
<table>
<thead>
<tr>
<th>Message Format</th>
<th>Message Format</th>
<th>2822.txt</th>
</tr>
</thead>
</table>
3 Requirements

3.1 Relevant Libraries

The libraries listed in Table 3-1 shall be available on a Linux Standard Base system, with the specified runtime names. The libraries listed in Table 3-2 are architecture specific, but shall be available on all LSB conforming systems. This list may be supplemented or amended by the relevant architecture specific part of the LSB Core Specification.

Table 3-1 Standard Library Names

<table>
<thead>
<tr>
<th>Library</th>
<th>Runtime Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>libcrypt</td>
<td>libcrypt.so.1</td>
</tr>
<tr>
<td>libdl</td>
<td>libdl.so.2</td>
</tr>
<tr>
<td>libgcc_s</td>
<td>libgcc_s.so.1</td>
</tr>
<tr>
<td>libncurses</td>
<td>libncurses.so.5</td>
</tr>
<tr>
<td>libncursesw</td>
<td>libncursesw.so.5</td>
</tr>
<tr>
<td>libnspr4</td>
<td>libnspr4.so</td>
</tr>
<tr>
<td>libnss3</td>
<td>libnss3.so</td>
</tr>
<tr>
<td>libpam</td>
<td>libpam.so.0</td>
</tr>
<tr>
<td>libpthread</td>
<td>libpthread.so.0</td>
</tr>
<tr>
<td>librt</td>
<td>librt.so.1</td>
</tr>
<tr>
<td>libssl3</td>
<td>libssl3.so</td>
</tr>
<tr>
<td>libstdcxx</td>
<td>libstdcxx.so.6</td>
</tr>
<tr>
<td>libutil</td>
<td>libutil.so.1</td>
</tr>
<tr>
<td>libz</td>
<td>libz.so.1</td>
</tr>
</tbody>
</table>

Table 3-2 Standard Library Names defined in the Architecture Specific Parts of the LSB Core Specification

<table>
<thead>
<tr>
<th>Library</th>
<th>Runtime Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>libc</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>libm</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>proginterp</td>
<td>See architecture specific part.</td>
</tr>
</tbody>
</table>

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.
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,

| forkpty(GLIBC_2.0) [SUSv4]

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 this module specification only. In the generic part, they will appear without symbol versions.
6 Relationship To ISO/IEC 9945 POSIX

This specification includes many interfaces described in POSIX 1003.1-2008 (ISO/IEC 9945-2009). Unless otherwise specified, such interfaces should behave exactly as described in that specification. Any conflict between the requirements described here and the POSIX 1003.1-2008 (ISO/IEC 9945-2009) standard is unintentional, except as explicitly noted otherwise.

**Note:** In addition to the differences noted in this specification, a report, ISO/IEC TR 24715-Technical Report on the Conflicts Between the ISO/IEC 9945 (POSIX) Standard and the Linux Standard Base Specification (LSB), identifies the differences between edition 3.1 of this specification and POSIX 1003.1-2001 (ISO/IEC 9945-2003) (more precisely, POSIX 2001 plus the first two corrigenda, informally known as the 2004 edition). It is the long-term plan of the Linux Foundation to converge the LSB Core specification with the ISO/IEC POSIX specification.

The LSB Specification Authority is responsible for deciding the meaning of conformance to normative referenced standards in the LSB context. Problem reports regarding underlying or referenced standards in any other context will be referred to the relevant maintenance body for that standard.
7 Relationship To Other Linux Foundation Specifications

The LSB is the base for several other specification projects under the umbrella of the Linux Foundation (LF). This specification is the foundation, and other specifications build on the interfaces defined here. However, beyond those specifications listed as Normative References, this specification has no dependencies on other LF projects.

ISO/IEC 23360 corresponds to an earlier edition of this specification (version 3.1), published as an ISO/IEC standard in 2006 after submission by the Linux Foundation. The ISO edition is also the subject of the technical report ISO/IEC TR 24715 referenced in the previous chapter.
II Executable And Linking Format (ELF)
8 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 is intended to document additions made since the publication of that document.
9 Low Level System Information

9.1 Operating System Interface

LSB-conforming applications shall assume that stack, heap and other allocated memory regions will be non-executable. The application must take steps to make them executable if needed.

9.2 Machine Interface

9.2.1 Data Representation

LSB-conforming applications shall use the data representation as defined in the Architecture specific ELF documents.

9.2.1.1 Fundamental Types

In addition to the fundamental types specified in the relevant architecture specific part of the LSB Core Specification, a 1 byte data type is defined here.

<table>
<thead>
<tr>
<th>Type</th>
<th>C</th>
<th>C++</th>
<th>sizeof</th>
<th>Alignment (bytes)</th>
<th>Architecture Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integral</td>
<td>_Bool</td>
<td>bool</td>
<td>1</td>
<td>1</td>
<td>byte</td>
</tr>
</tbody>
</table>
10 Object Format

10.1 Object Files

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
- the relevant architecture specific ABI supplement.
- this specification
- the relevant architecture specific part of the LSB Core Specification

Conforming implementations may also support other unspecified object file formats.

10.2 Sections

10.2.1 Introduction

As described in System V ABI, an ELF object file contains a number of sections.

10.2.2 Sections Types

The section header table is an array of Elf32_Shdr or Elf64_Shdr structures as described in System V ABI. The sh_type member shall be either a value from Table 10-1, drawn from the System V ABI, or one of the additional values specified in Table 10-2.

A section header's sh_type member specifies the section's semantics.

10.2.2.1 ELF Section Types

The following section types are defined in the System V ABI and the System V ABI Update.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHT_DYNAMIC</td>
<td>0x6</td>
<td>The section holds information for dynamic linking. Currently, an object file shall have only one dynamic section, but this restriction may be relaxed in the future. See 'Dynamic Section' in Chapter 5 of System V ABI Update for details.</td>
</tr>
<tr>
<td>SHT_DYNSYM</td>
<td>0xb</td>
<td>This section holds a minimal set of symbols adequate for dynamic linking. See also SHT_SYMTAB. Currently, an object file may have either a section of SHT_SYMTAB type or a section of SHT_DYNSYM type, but not both.</td>
</tr>
<tr>
<td>Section Type</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SHT_FINI_ARRAY</td>
<td>0xf</td>
<td>This section contains an array of pointers to termination functions, as described in 'Initialization and Termination Functions' in Chapter 5 of System V ABI Update. Each pointer in the array is taken as a parameterless procedure with a void return.</td>
</tr>
<tr>
<td>SHT_HASH</td>
<td>0x5</td>
<td>The section holds a symbol hash table. Currently, an object file shall have only one hash table, but this restriction may be relaxed in the future. See 'Hash Table' in Chapter 5 of System V ABI Update for details.</td>
</tr>
<tr>
<td>SHT_INIT_ARRAY</td>
<td>0xe</td>
<td>This section contains an array of pointers to initialization functions, as described in 'Initialization and Termination Functions' in Chapter 5 of System V ABI Update. Each pointer in the array is taken as a parameterless procedure with a void return.</td>
</tr>
<tr>
<td>SHT_NOBITS</td>
<td>0x8</td>
<td>A section of this type occupies no space in the file but otherwise resembles SHT_PROGBITS. Although this section contains no bytes, the sh_offset member contains the conceptual file offset.</td>
</tr>
<tr>
<td>SHT_NOTE</td>
<td>0x7</td>
<td>The section holds information that marks the file in some way. See `Note Section' in Chapter 5 of System V ABI Update for details.</td>
</tr>
<tr>
<td>SHT_NULL</td>
<td>0x0</td>
<td>This value marks the section header as inactive; it does not have an associated section. Other members of the section header have undefined values.</td>
</tr>
<tr>
<td>SHT_PREINIT_ARRAY</td>
<td>0x10</td>
<td>This section contains an array of pointers to functions that are invoked be-</td>
</tr>
</tbody>
</table>
fore all other initialization functions, as described in `Initialization and Termination Functions' in Chapter 5 of System V ABI Update. Each pointer in the array is taken as a parameterless procedure with a void return.

<table>
<thead>
<tr>
<th>Section</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHT_PROGBITS</td>
<td>0x1</td>
<td>The section holds information defined by the program, whose format and meaning are determined solely by the program.</td>
</tr>
<tr>
<td>SHT_REL</td>
<td>0x9</td>
<td>The section holds relocation entries without explicit addends, such as type Elf32_Rel for the 32-bit class of object files or type Elf64_Rel for the 64-bit class of object files. An object file may have multiple relocation sections. See 'Relocation' in Chapter 4 of System V ABI Update for details.</td>
</tr>
<tr>
<td>SHT_RELA</td>
<td>0x4</td>
<td>The section holds relocation entries with explicit addends, such as type Elf32_Rela for the 32-bit class of object files or type Elf64_Rela for the 64-bit class of object files. An object file may have multiple relocation sections. See 'Relocation' in Chapter 4 of System V ABI Update for details.</td>
</tr>
<tr>
<td>SHT_STRTAB</td>
<td>0x3</td>
<td>The section holds a string table. An object file may have multiple string table sections. See 'String Table' in Chapter 4 of System V ABI Update for details.</td>
</tr>
</tbody>
</table>
| SHT_SYMTAB  | 0x2   | This section holds a symbol table. Currently, an object file may have either a section of SHT_SYMTAB type or a section of SHT_DYN-SYM type, but not both. This restriction may be relaxed in the future. Typically, SHT_SYMTAB provides symbols for link
10.2.2.2 Additional Section Types

The following additional section types are defined here.

Table 10-2 Additional Section Types

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHT_GNU_verdef</td>
<td>0x6ffffffd</td>
<td>This section contains the symbol versions that are provided.</td>
</tr>
<tr>
<td>SHT_GNU_verneed</td>
<td>0x6ffffffe</td>
<td>This section contains the symbol versions that are required.</td>
</tr>
<tr>
<td>SHT_GNU_versym</td>
<td>0x6fffffff</td>
<td>This section contains the Symbol Version Table.</td>
</tr>
</tbody>
</table>

10.3 Special Sections

10.3.1 Special Sections

Various sections hold program and control information. Sections in the lists below are used by the system and have the indicated types and attributes.

10.3.1.1 ELF Special Sections

The following sections are defined in the System V ABI and the System V ABI Update.

Table 10-3 ELF Special Sections

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>.bss</td>
<td>SHT_NOBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.comment</td>
<td>SHT_PROGBITS</td>
<td>SHF_MERGE+SHF_HEADERS</td>
</tr>
<tr>
<td>.data</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.data1</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.debug</td>
<td>SHT_PROGBITS</td>
<td>0</td>
</tr>
<tr>
<td>.dynamic</td>
<td>SHT_DYNTAB</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.dynstr</td>
<td>SHT_STRTAB</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.dynsym</td>
<td>SHT_DYNSYM</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.fini</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_EXECINSTR</td>
</tr>
<tr>
<td>.fini_array</td>
<td>SHT_FINI_ARRAY</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.hash</td>
<td>SHT_HASH</td>
<td>SHF_ALLOC</td>
</tr>
</tbody>
</table>
.init
SHT_PROGBITS SHF_ALLOC+SHF_EX-ECINSTR

.init_array
SHT_INIT_ARRAY SHF_ALLOC+SHF_WRITE

.interp
SHT_PROGBITS SHF_ALLOC

.line
SHT_PROGBITS 0

.note
SHT_NOTE 0

.preinit_array
SHT_PREINIT_ARRAY SHF_ALLOC+SHF_WRITE

.rodata
SHT_PROGBITS SHF_ALLOC+SHF_MERGE+SHF_STRINGS

.rodata1
SHT_PROGBITS SHF_ALLOC+SHF_MERGE+SHF_STRINGS

.shstrtab
SHT_STRTAB 0

.strtab
SHT_STRTAB SHF_ALLOC

.symtab
SHT_SYMTAB SHF_ALLOC

.tbss
SHT_NOBITS SHF_ALLOC+SHF_WRITE+SHF_TLS

.tdata
SHT_PROGBITS SHF_ALLOC+SHF_WRITE+SHF_TLS

.text
SHT_PROGBITS SHF_ALLOC+SHF_EX-ECINSTR

.bss
This section holds data that contributes to the program's memory image. The pro-
gram may treat this data as uninitialized. However, the system shall initialize this
data with zeroes when the program begins to run. The section occupies no file
space, as indicated by the section type, SHT_NOBITS.

.comment
This section holds version control information.

.data
This section holds initialized data that contribute to the program's memory image.

.data1
This section holds initialized data that contribute to the program's memory image.

.debug
This section holds information for symbolic debugging. The contents are unspe-
cified. All section names with the prefix .debug hold information for symbolic de-
bugging. The contents of these sections are unspecified.

.dynamic
This section holds dynamic linking information. The section's attributes will in-
clude the SHF_ALLOC bit. Whether the SHF_WRITE bit is set is processor spe-
cific. See Chapter 5 of System V ABI Update for more information.

.dynstr
This section holds strings needed for dynamic linking, most commonly the strings
that represent the names associated with symbol table entries. See Chapter 5 of
.dynsym

This section holds the dynamic linking symbol table, as described in `Symbol Table' of System V ABI Update.

.fini

This section holds executable instructions that contribute to the process termination code. That is, when a program exits normally, the system arranges to execute the code in this section.

.fini_array

This section holds an array of function pointers that contributes to a single termination array for the executable or shared object containing the section.

.hash

This section holds a symbol hash table. See `Hash Table' in Chapter 5 of System V ABI Update for more information.

.init

This section holds executable instructions that contribute to the process initialization code. When a program starts to run, the system arranges to execute the code in this section before calling the main program entry point (called main for C programs).

.init_array

This section holds an array of function pointers that contributes to a single initialization array for the executable or shared object containing the section.

.interp

This section holds the path name of a program interpreter. If the file has a loadable segment that includes relocation, the sections' attributes will include the SHF_ALLOC bit; otherwise, that bit will be off. See Chapter 5 of System V ABI Update for more information.

.line

This section holds line number information for symbolic debugging, which describes the correspondence between the source program and the machine code. The contents are unspecified.

.note

This section holds information in the format that `Note Section' in Chapter 5 of System V ABI Update describes.

.preinit_array

This section holds an array of function pointers that contributes to a single pre-initialization array for the executable or shared object containing the section.

.rodata

This section holds read-only data that typically contribute to a non-writable segment in the process image. See `Program Header' in Chapter 5 of System V ABI Update for more information.

.rodata1

This section holds read-only data that typically contribute to a non-writable seg-

© 2015 Linux Foundation
ment in the process image. See `Program Header' in Chapter 5 of System V ABI Update for more information.

`.shstrtab
This section holds section names.

`.strtab
This section holds strings, most commonly the strings that represent the names associated with symbol table entries. If the file has a loadable segment that includes the symbol string table, the section's attributes will include the SHF_ALLOC bit; otherwise, that bit will be off.

`.symtab
This section holds a symbol table, as `Symbol Table’ in Chapter 4 of System V ABI Update describes. If the file has a loadable segment that includes the symbol table, the section's attributes will include the SHF_ALLOC bit; otherwise, that bit will be off.

`.tbss
This section holds uninitialized thread-local data that contribute to the program's memory image. By definition, the system initializes the data with zeros when the data is instantiated for each new execution flow. The section occupies no file space, as indicated by the section type, SHT_NOBITS. Implementations need not support thread-local storage.

`.tdata
This section holds initialized thread-local data that contributes to the program's memory image. A copy of its contents is instantiated by the system for each new execution flow. Implementations need not support thread-local storage.

`.text
This section holds the `text', or executable instructions, of a program.

10.3.1.2 Additional Special Sections
Object files in an LSB conforming application may also contain one or more of the additional special sections described below.

Table 10-4 Additional Special Sections

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>.ctors</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.data.rel.ro</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.dtors</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.eh_frame</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.eh_frame_hdr</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.gcc_except_table</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.gnu.version</td>
<td>SHT_GNU_versym</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.gnu.version_d</td>
<td>SHT_GNU_verdef</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.gnu.version_r</td>
<td>SHT_GNU_verneed</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.got.plt</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
</tbody>
</table>
### .ctors
This section contains a list of global constructor function pointers.

### .data.rel.ro
This section holds initialized data that contribute to the program's memory image. This section may be made read-only after relocations have been applied.

### .dtors
This section contains a list of global destructor function pointers.

### .eh_frame
This section contains information necessary for frame unwinding during exception handling. See Section 10.6.1.

### .eh_frame_hdr
This section contains a pointer to the .eh_frame section which is accessible to the runtime support code of a C++ application. This section may also contain a binary search table which may be used by the runtime support code to more efficiently access records in the .eh_frame section. See Section 10.6.2.

### .gcc_except_table
This section holds Language Specific Data.

### .gnu.version
This section contains the Symbol Version Table. See Section 10.7.2.

### .gnu.version_d
This section contains the Version Definitions. See Section 10.7.3.

### .gnu.version_r
This section contains the Version Requirements. See Section 10.7.4.

### .got.plt
This section holds the read-only portion of the GLobal Offset Table. This section may be made read-only after relocations have been applied.

### .jcr
This section contains information necessary for registering compiled Java classes. The contents are compiler-specific and used by compiler initialization functions.

### .note.ABI-tag
Specify ABI details. See Section 10.8.

### .stab
This section contains debugging information. The contents are not specified as part of the LSB.

---

<table>
<thead>
<tr>
<th>Section</th>
<th>Type</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>.jcr</td>
<td>SHT_PROGBITS</td>
<td>SHF_ALLOC+SHF_WRITE</td>
</tr>
<tr>
<td>.note.ABI-tag</td>
<td>SHT_NOTE</td>
<td>SHF_ALLOC</td>
</tr>
<tr>
<td>.stab</td>
<td>SHT_PROGBITS</td>
<td>0</td>
</tr>
<tr>
<td>.stabstr</td>
<td>SHT_STRTAB</td>
<td>0</td>
</tr>
</tbody>
</table>
.stabstr

This section contains strings associated with the debugging information contained in the .stab section.

10.4 Symbol Mapping

10.4.1 Introduction

Symbols in a source program are translated by the compilation system into symbols that exist in the object file.

10.4.1.1 C Language

External C symbols shall be unchanged in an object file's symbol table.

10.5 DWARF Extensions

The LSB does not specify debugging information, however, some additional sections contain information which is encoded using the encoding as specified by DWARF Debugging Information Format, Version 4 with extensions defined here.

10.5.1 DWARF Exception Header Encoding

The DWARF Exception Header Encoding is used to describe the type of data used in the .eh_frame and .eh_frame_hdr section. The upper 4 bits indicate how the value is to be applied. The lower 4 bits indicate the format of the data.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW_EH_PE_absptr</td>
<td>0x00</td>
<td>The Value is a literal pointer whose size is determined by the architecture.</td>
</tr>
<tr>
<td>DW_EH_PE_uleb128</td>
<td>0x01</td>
<td>Unsigned value is encoded using the Little Endian Base 128 (LEB128) as defined by DWARF Debugging Information Format, Version 4.</td>
</tr>
<tr>
<td>DW_EH_PE_udata2</td>
<td>0x02</td>
<td>A 2 bytes unsigned value.</td>
</tr>
<tr>
<td>DW_EH_PE_udata4</td>
<td>0x03</td>
<td>A 4 bytes unsigned value.</td>
</tr>
<tr>
<td>DW_EH_PE_udata8</td>
<td>0x04</td>
<td>An 8 bytes unsigned value.</td>
</tr>
<tr>
<td>DW_EH_PE_sleb128</td>
<td>0x09</td>
<td>Signed value is encoded using the Little Endian Base 128 (LEB128) as defined by DWARF Debugging Information Format, Version 4.</td>
</tr>
<tr>
<td>DW_EH_PE_sdata2</td>
<td>0x0A</td>
<td>A 2 bytes signed value.</td>
</tr>
<tr>
<td>DW_EH_PE_sdata4</td>
<td>0x0B</td>
<td>A 4 bytes signed value.</td>
</tr>
<tr>
<td>DW_EH_PE_sdata8</td>
<td>0x0C</td>
<td>An 8 bytes signed value.</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
### Table 10-6 DWARF Exception Header application

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW_EH_PE_pcrel</td>
<td>0x10</td>
<td>Value is relative to the current program counter.</td>
</tr>
<tr>
<td>DW_EH_PE_textrel</td>
<td>0x20</td>
<td>Value is relative to the beginning of the .text section.</td>
</tr>
<tr>
<td>DW_EH_PE_databrel</td>
<td>0x30</td>
<td>Value is relative to the beginning of the .got or .eh_frame_hdr section.</td>
</tr>
<tr>
<td>DW_EH_PE_funcrel</td>
<td>0x40</td>
<td>Value is relative to the beginning of the function.</td>
</tr>
<tr>
<td>DW_EH_PE_aligned</td>
<td>0x50</td>
<td>Value is aligned to an address unit sized boundary.</td>
</tr>
</tbody>
</table>

One special encoding, 0xff (DW_EH_PE_omit), shall be used to indicate that no value is present.

### 10.5.2 DWARF CFI Extensions

In addition to the Call Frame Instructions defined in section 6.4.2 of [DWARF Debugging Information Format, Version 4](https://www.linuxfoundation.org/en/oss/dwarf-architecture), the following additional Call Frame Instructions may also be used.

### Table 10-7 Additional DWARF Call Frame Instructions

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW_CFA_GNU_args_size</td>
<td>0x2e</td>
<td>The DW_CFA_GNU_args_size instruction takes an unsigned LEB128 operand representing an argument size. This instruction specifies the total of the size of the arguments which have been pushed onto the stack.</td>
</tr>
<tr>
<td>DW_CFA_GNU_negative_offset_extended</td>
<td>0x2f</td>
<td>The DW_CFA_def_cfa_sf instruction takes two operands: an unsigned LEB128 value representing a register number and an unsigned LEB128 which represents the magnitude of the offset. This instruction is identical to DW_CFA_offset_extended_sf except that the operand is subtracted to produce the offset. This instructions is obsoleted by DW_CFA_offset_extended.</td>
</tr>
</tbody>
</table>
10.6 Exception Frames

When using languages that support exceptions, such as C++, additional information must be provided to the runtime environment that describes the call frames that must be unwound during the processing of an exception. This information is contained in the special sections `.eh_frame` and `.eh_framehdr`.

Note: The format of the `.eh_frame` section is similar in format and purpose to the `.debug_frame` section which is specified in DWARF Debugging Information Format, Version 4. Readers are advised that there are some subtle difference, and care should be taken when comparing the two sections.

10.6.1 The `.eh_frame` section

The `.eh_frame` section shall contain 1 or more Call Frame Information (CFI) records. The number of records present shall be determined by size of the section as contained in the section header. Each CFI record contains a Common Information Entry (CIE) record followed by 1 or more Frame Description Entry (FDE) records. Both CIEs and FDEs shall be aligned to an addressing unit sized boundary.

Table 10-8 Call Frame Information Format

| Common Information Entry Record | Frame Description Entry Record(s) |

10.6.1.1 The Common Information Entry Format

Table 10-9 Common Information Entry Format

<table>
<thead>
<tr>
<th>Length</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extended Length</td>
<td>Optional</td>
</tr>
<tr>
<td>CIE ID</td>
<td>Required</td>
</tr>
<tr>
<td>Version</td>
<td>Required</td>
</tr>
<tr>
<td>Augmentation String</td>
<td>Required</td>
</tr>
<tr>
<td>Code Alignment Factor</td>
<td>Required</td>
</tr>
<tr>
<td>Data Alignment Factor</td>
<td>Required</td>
</tr>
<tr>
<td>Return Address Register</td>
<td>Required</td>
</tr>
<tr>
<td>Augmentation Data Length</td>
<td>Optional</td>
</tr>
<tr>
<td>Augmentation Data</td>
<td>Optional</td>
</tr>
<tr>
<td>Initial Instructions</td>
<td>Required</td>
</tr>
<tr>
<td>Padding</td>
<td></td>
</tr>
</tbody>
</table>

Length

A 4 byte unsigned value indicating the length in bytes of the CIE structure, not including the Length field itself. If Length contains the value 0xffffffff, then the length is contained in the Extended Length field. If Length contains the value 0, then this CIE shall be considered a terminator and processing shall end.

Extended Length

A 8 byte unsigned value indicating the length in bytes of the CIE structure, not including the Length and Extended Length fields themselves. This field is not present unless the Length field contains the value 0xffffffff.
CIE ID
A 4 byte unsigned value that is used to distinguish CIE records from FDE records. This value shall always be 0, which indicates this record is a CIE.

Version
A 1 byte value that identifies the version number of the frame information structure. This value shall be 1.

Augmentation String
This value is a NUL terminated string that identifies the augmentation to the CIE or to the FDEs associated with this CIE. A zero length string indicates that no augmentation data is present. The augmentation string is case sensitive and shall be interpreted as described below.

Code Alignment Factor
An unsigned LEB128 encoded value that is factored out of all advance location instructions that are associated with this CIE or its FDEs. This value shall be multiplied by the delta argument of an advance location instruction to obtain the new location value.

Data Alignment Factor
A signed LEB128 encoded value that is factored out of all offset instructions that are associated with this CIE or its FDEs. This value shall be multiplied by the register offset argument of an offset instruction to obtain the new offset value.

Augmentation Length
An unsigned LEB128 encoded value indicating the length in bytes of the Augmentation Data. This field is only present if the Augmentation String contains the character ‘z’.

Augmentation Data
A block of data whose contents are defined by the contents of the Augmentation String as described below. This field is only present if the Augmentation String contains the character ‘z’. The size of this data is given by the Augmentation Length.

Initial Instructions
Initial set of Call Frame Instructions. The number of instructions is determined by the remaining space in the CIE record.

Padding
Extra bytes to align the CIE structure to an addressing unit size boundary.

10.6.1.1 Augmentation String Format
The Augmentation String indicates the presence of some optional fields, and how those fields should be interpreted. This string is case sensitive. Each character in the augmentation string in the CIE can be interpreted as below:

‘z’
A ‘z’ may be present as the first character of the string. If present, the Augmentation Data field shall be present. The contents of the Augmentation Data shall be interpreted according to other characters in the Augmentation String.

‘L’
A ‘L’ may be present at any position after the first character of the string. This character may only be present if ‘z’ is the first character of the string. If present, it
indicates the presence of one argument in the Augmentation Data of the CIE, and a corresponding argument in the Augmentation Data of the FDE. The argument in the Augmentation Data of the CIE is 1-byte and represents the pointer encoding used for the argument in the Augmentation Data of the FDE, which is the address of a language-specific data area (LSDA). The size of the LSDA pointer is specified by the pointer encoding used.

'P'

A 'P' may be present at any position after the first character of the string. This character may only be present if 'z' is the first character of the string. If present, it indicates the presence of two arguments in the Augmentation Data of the CIE. The first argument is 1-byte and represents the pointer encoding used for the second argument, which is the address of a personality routine handler. The personality routine is used to handle language and vendor-specific tasks. The system unwind library interface accesses the language-specific exception handling semantics via the pointer to the personality routine. The personality routine does not have an ABI-specific name. The size of the personality routine pointer is specified by the pointer encoding used.

'R'

A 'R' may be present at any position after the first character of the string. This character may only be present if 'z' is the first character of the string. If present, The Augmentation Data shall include a 1 byte argument that represents the pointer encoding for the address pointers used in the FDE.

### 10.6.1.2 The Frame Description Entry Format

<table>
<thead>
<tr>
<th>Table 10-10 Frame Description Entry Format</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Length</strong></td>
</tr>
<tr>
<td><strong>Extended Length</strong></td>
</tr>
<tr>
<td><strong>FDE Pointer</strong></td>
</tr>
<tr>
<td><strong>PC Begin</strong></td>
</tr>
<tr>
<td><strong>PC Range</strong></td>
</tr>
<tr>
<td><strong>Augmentation Data Length</strong></td>
</tr>
<tr>
<td><strong>Augmentation Data</strong></td>
</tr>
<tr>
<td><strong>Call Frame Instructions</strong></td>
</tr>
<tr>
<td><strong>Padding</strong></td>
</tr>
</tbody>
</table>

**Length**

A 4 byte unsigned value indicating the length in bytes of the FDE structure, not including the Length field itself. If Length contains the value 0xffffffff, then the length is contained the Extended Length field. If Length contains the value 0, then this FDE shall be considered a terminator and processing shall end.

**Extended Length**

A 8 byte unsigned value indicating the length in bytes of the FDE structure, not including the Length or Extended Length field themselves. This field is not present unless the Length field contains the value 0xffffffff.

**CIE Pointer**

A 4 byte unsigned value that when subtracted from the offset of the CIE Pointer in the current FDE yields the offset of the start of the associated CIE. This value shall never be 0.
PC Begin
An encoded value that indicates the address of the initial location associated with this FDE. The encoding format is specified in the Augmentation Data.

PC Range
An absolute value that indicates the number of bytes of instructions associated with this FDE.

Augmentation Length
An unsigned LEB128 encoded value indicating the length in bytes of the Augmentation Data. This field is only present if the Augmentation String in the associated CIE contains the character 'z'.

Augmentation Data
A block of data whose contents are defined by the contents of the Augmentation String in the associated CIE as described above. This field is only present if the Augmentation String in the associated CIE contains the character 'z'. The size of this data is given by the Augmentation Length.

Call Frame Instructions
A set of Call Frame Instructions.

Padding
Extra bytes to align the FDE structure to an addressing unit size boundary.

10.6.2 The .eh_frame_hdr section
The .eh_frame_hdr section contains additional information about the .eh_frame section. A pointer to the start of the .eh_frame data, and optionally, a binary search table of pointers to the .eh_frame records are found in this section.

Data in this section is encoded according to Section 10.5.1.

<table>
<thead>
<tr>
<th>Encoding Field</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>unsigned byte</td>
<td>version</td>
</tr>
<tr>
<td>unsigned byte</td>
<td>eh_frame_ptr_enc</td>
</tr>
<tr>
<td>unsigned byte</td>
<td>fde_count_enc</td>
</tr>
<tr>
<td>unsigned byte</td>
<td>table_enc</td>
</tr>
<tr>
<td>encoded</td>
<td>eh_frame_ptr</td>
</tr>
<tr>
<td>encoded</td>
<td>fde_count</td>
</tr>
<tr>
<td></td>
<td>binary search table</td>
</tr>
</tbody>
</table>

version
Version of the .eh_frame_hdr format. This value shall be 1.

eh_frame_ptr_enc
The encoding format of the eh_frame_ptr field.

fde_count_enc
The encoding format of the fde_count field. A value of DW_EH_PE_omit indicates the binary search table is not present.
The encoding format of the entries in the binary search table. A value of DW_EH_PE_omit indicates the binary search table is not present.

The encoded value of the pointer to the start of the .eh_frame section.

The encoded value of the count of entries in the binary search table.

A binary search table containing fde_count entries. Each entry of the table consist of two encoded values, the initial location, and the address. The entries are sorted in an increasing order by the initial location value.

10.7 Symbol Versioning

10.7.1 Introduction

This chapter describes the Symbol Versioning mechanism. All ELF objects may provide or depend on versioned symbols. Symbol Versioning is implemented by 3 section types: SHT_GNU_versym, SHT_GNU_verdef, and SHT_GNU_verneed.

The prefix Elfxx in the following descriptions and code fragments stands for either "Elf32" or "Elf64", depending on the architecture.

Versions are described by strings. The structures that are used for symbol versions also contain a member that holds the ELF hashing values of the strings. This allows for more efficient processing.

10.7.2 Symbol Version Table

The special section .gnu.version which has a section type of SHT_GNU_versym shall contain the Symbol Version Table. This section shall have the same number of entries as the Dynamic Symbol Table in the .dynsym section.

The .gnu.version section shall contain an array of elements of type Elfxx_Half. Each entry specifies the version defined for or required by the corresponding symbol in the Dynamic Symbol Table.

The values in the Symbol Version Table are specific to the object in which they are located. These values are identifiers that are provided by the the vna_other member of the Elfxx_Vernaux structure or the vd_ndx member of the Elfxx_Verdef structure.

The values 0 and 1 are reserved.

0

The symbol is local, not available outside the object.

1

The symbol is defined in this object and is globally available.

All other values are used to identify version strings located in one of the other Symbol Version sections. The value itself is not the version associated with the symbol. The string identified by the value defines the version of the symbol.

10.7.3 Version Definitions

The special section .gnu.version_d which has a section type of SHT_GNU_verdef shall contain symbol version definitions. The number of entries in this section shall be
contained in the DT_VERDEFNUM entry of the Dynamic Section .dynamic. The sh_link member of the section header (see figure 4-8 in the System V ABI) shall point to the section that contains the strings referenced by this section.

The section shall contain an array of Elfxx_Verdef structures, as described in Figure 10-1, optionally followed by an array of Elfxx_Verdaux structures, as defined in Figure 10-2.

```c
typedef struct {
    Elfxx_Half vd_version;
    Elfxx_Half vd_flags;
    Elfxx_Half vd_ndx;
    Elfxx_Half vd_cnt;
    Elfxx_Word vd_hash;
    Elfxx_Word vd_aux;
    Elfxx_Word vd_next;
} Elfxx_Verdef;
```

**Figure 10-1 Version Definition Entries**

- **vd_version**  
  Version revision. This field shall be set to 1.

- **vd_flags**  
  Version information flag bitmask.

- **vd_ndx**  
  Version index numeric value referencing the SHT_GNU_versym section.

- **vd_cnt**  
  Number of associated verdaux array entries.

- **vd_hash**  
  Version name hash value (ELF hash function).

- **vd_aux**  
  Offset in bytes to a corresponding entry in an array of Elfxx_Verdaux structures as defined in Figure 10-2.

- **vd_next**  
  Offset to the next verdef entry, in bytes.

```c
typedef struct {
    Elfxx_Word vda_name;
    Elfxx_Word vda_next;
} Elfxx_Verdaux;
```

**Figure 10-2 Version Definition Auxiliary Entries**

- **vda_name**  
  Offset to the version or dependency name string in the section header, in bytes.

- **vda_next**  
  Offset to the next verdaux entry, in bytes.

### 10.7.4 Version Requirements

The special section .gnu.version_r which has a section type of SHT_GNU_verneed shall contain required symbol version definitions. The number of entries in this section
shall be contained in the DT_VERNEEDNUM entry of the Dynamic Section .dynamic. The sh_link member of the section header (see figure 4-8 in System V ABI) shall point to the section that contains the strings referenced by this section.

The section shall contain an array of Elfxx_Verneed structures, as described in Figure 10-3, optionally followed by an array of Elfxx_Vernaux structures, as defined in Figure 10-4.

typedef struct {
  Elfxx_Half vn_version;
  Elfxx_Half vn_cnt;
  Elfxx_Word vn_file;
  Elfxx_Word vn_aux;
  Elfxx_Word vn_next;
} Elfxx_Verneed;

Figure 10-3 Version Needed Entries

vn_version
  Version of structure. This value is currently set to 1, and will be reset if the
  versioning implementation is incompatibly altered.

vn_cnt
  Number of associated verneed array entries.

vn_file
  Offset to the file name string in the section header, in bytes.

vn_aux
  Offset to a corresponding entry in the vernaux array, in bytes.

vn_next
  Offset to the next verneed entry, in bytes.

typedef struct {
  Elfxx_Word vna_hash;
  Elfxx_Half vna_flags;
  Elfxx_Half vna_other;
  Elfxx_Word vna_name;
  Elfxx_Word vna_next;
} Elfxx_Vernaux;

Figure 10-4 Version Needed Auxiliary Entries

vna_hash
  Dependency name hash value (ELF hash function).

vna_flags
  Dependency information flag bitmask.

vna_other
  Object file version identifier used in the .gnu.version symbol version array. Bit
  number 15 controls whether or not the object is hidden; if this bit is set, the object
  cannot be used and the static linker will ignore the symbol’s presence in the object.

vna_name
  Offset to the dependency name string in the section header, in bytes.
10.7.5 Startup Sequence

When loading a sharable object the system shall analyze version definition data from the loaded object to assure that it meets the version requirements of the calling object. This step is referred to as definition testing. The dynamic loader shall retrieve the entries in the caller’s Elfxx_Verneed array and attempt to find matching definition information in the loaded Elfxx_Verdef table.

Each object and dependency shall be tested in turn. If a symbol definition is missing and the vna_flags bit for VER_FLG_WEAK is not set, the loader shall return an error and exit. If the vna_flags bit for VER_FLG_WEAK is set in the Elfxx_Vernaux entry, and the loader shall issue a warning and continue operation.

When the versions referenced by undefined symbols in the loaded object are found, version availability is certified. The test completes without error and the object shall be made available.

10.7.6 Symbol Resolution

When symbol versioning is used in an object, relocations extend definition testing beyond the simple match of symbol name strings: the version of the reference shall also equal the name of the definition.

The same index that is used in the symbol table can be referenced in the SHT_GNU_versym section, and the value of this index is then used to acquire name data. The corresponding requirement string is retrieved from the Elfxx_Verneed array, and likewise, the corresponding definition string from the Elfxx_Verdef table.

If the high order bit (bit number 15) of the version symbol is set, the object cannot be used and the static linker shall ignore the symbol's presence in the object.

When an object with a reference and an object with the definition are being linked, the following rules shall govern the result:

• The object with the reference and the object with the definitions both use versioning. All described matching is processed in this case. A fatal error shall be triggered when no matching definition can be found in the object whose name is the one referenced by the vn_name element in the Elfxx_Verneed entry.

• The object with the reference does not use versioning, while the object with the definitions does. In this instance, only the definitions with index numbers 1 and 2 will be used in the reference match, the same identified by the static linker as the base definition. In cases where the static linker was not used, such as in calls to dlopen(), a version that does not have the base definition index shall be acceptable if it is the only version for which the symbol is defined.

• The object with the reference uses versioning, but the object with the definitions specifies none. A matching symbol shall be accepted in this case. A fatal error shall be triggered if a corruption in the required symbols list obscures an outdated object file and causes a match on the object filename in the Elfxx_Verneed entry.

• Neither the object with the reference nor the object with the definitions use versioning. The behavior in this instance shall default to pre-existing symbol rules.

10.8 ABI note tag

Every executable shall contain a section named .note.ABI-tag of type SHT_NOTE. This section is structured as a note section as documented in the ELF spec. The section shall contain at least the following entry. The name field (namesz/name) contains the string "GNU". The type field shall be 1. The descsz field shall be at least 16, and the
first 16 bytes of the desc field shall be as follows.
The first 32-bit word of the desc field shall be 0 (this signifies a Linux executable). The second, third, and fourth 32-bit words of the desc field contain the earliest compatible kernel version. For example, if the 3 words are 2, 2, and 5, this signifies a 2.2.5 kernel.
11 Dynamic Linking

11.1 Program Loading and Dynamic Linking

LSB-conforming implementations shall support the object file information and system actions that create running programs as specified in the System V ABI and System V ABI Update and as further required by this specification and the relevant architecture specific part of the LSB Core Specification.

Any shared object that is loaded shall contain sufficient DT_NEEDED records to satisfy the symbols on the shared library.

11.2 Program Header

In addition to the Segment Types defined in the System V ABI and System V ABI Update the following Segment Types shall also be supported.

Table 11-1 Linux Segment Types

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PT_GNU_EH_FRAME</td>
<td>0x6474e550</td>
</tr>
<tr>
<td>PT_GNU_STACK</td>
<td>0x6474e551</td>
</tr>
<tr>
<td>PT_GNU_RELRO</td>
<td>0x6474e552</td>
</tr>
</tbody>
</table>

PT_GNU_EH_FRAME

The array element specifies the location and size of the exception handling information as defined by the .eh_frame_hdr section.

PT_GNU_STACK

The p_flags member specifies the permissions on the segment containing the stack and is used to indicate whether the stack should be executable. The absence of this header indicates that the stack will be executable.

PT_GNU_RELRO

The array element specifies the location and size of a segment which may be made read-only after relocations have been processed.

11.3 Dynamic Entries

11.3.1 Introduction

As described in System V ABI, if an object file participates in dynamic linking, its program header table shall have an element of type PT_DYNAMIC. This ‘segment’ contains the .dynamic section. A special symbol, _DYNAMIC, labels the section, which contains an array of the following structures.

```c
typedef struct {
    Elf32_Sword d_tag;
    union {
        Elf32_Word d_val;
        Elf32_Addr d_ptr;
    } d_un;
} Elf32_Dyn;

typedef struct {
    Elf64_Sxword d_tag;
} Elf64_Dyn;
extern Elf32_Dyn _DYNAMIC[];
```
union {
    Elf64_Xword     d_val;
    Elf64_Addr      d_ptr;
} d_un;
} Elf64_Dyn;

extern Elf64_Dyn        _DYNAMIC[];

Figure 11-1 Dynamic Structure

For each object with this type, \texttt{d\_tag} controls the interpretation of \texttt{d\_un}.

## 11.3.2 Dynamic Entries

### 11.3.2.1 ELF Dynamic Entries

The following dynamic entries are defined in the System V ABI and System V ABI Update.

- **DT_BIND_NOW**
  
  Process relocations of object

- **DT_DEBUG**
  
  For debugging; unspecified

- **DT_FINI**
  
  Address of termination function

- **DT_FINI_ARRAY**
  
  The address of an array of pointers to termination functions.

- **DT_FINI_ARRAYSZ**
  
  Size in bytes of DT_FINI_ARRAY

- **DT_FLAGS**
  
  Flag values specific to the object being loaded

- **DT_HASH**
  
  Address of symbol hash table

- **DT_HIPROC**
  
  End of processor-specific

- **DT_INIT**
  
  Address of init function

- **DT_INIT_ARRAY**
  
  The address of an array of pointers to initialization functions.

- **DT_INIT_ARRAYSZ**
  
  Size in bytes of DT_INIT_ARRAY

- **DT_JMPREL**
  
  Address of PLT relocations
DT_LOPROC
   Start of processor-specific

DT_NEEDED
   Name of needed library

DT_NULL
   Marks end of dynamic section

DT_PLTREL
   Type of reloc in PLT

DT_PLTRELSZ
   Size in bytes of PLT relocs

DT_PREINIT_ARRAY
   Array with addresses of preinit functions

DT_PREINIT_ARRAYSZ
   Size in bytes of DT_PREINIT_ARRAY

DT_REL
   Address of Rel relocs

DT_RELA
   Address of Rela relocs

DT_RELAENT
   Size of one Rela reloc

DT_RELASZ
   Total size of Rela relocs

DT_RELENT
   Size of one Rel reloc

DT_RELSZ
   Total size of Rel relocs

DT_RPATH
   Library search path

DT_RUNPATH
   null-terminated library search path string

DT_SONAME
   Name of shared object

DT_STRSZ
   Size of string table

DT_STRTAB
   Address of string table
DT_SYMBOLIC
Start symbol search here

DT_SYMMENT
Size of one symbol table entry

DT_SYMTAB
Address of symbol table

DT_TEXTREL
Reloc might modify .text

11.3.2.2 Additional Dynamic Entries
An LSB conforming object may also use the following additional Dynamic Entry types.

DT_ADDRRNGHI
Values from DT_ADDRRNGLO through DT_ADDRRNGHI are reserved for definition by an architecture specific part.

DT_ADDRRNGLO
Values from DT_ADDRRNGLO through DT_ADDRRNGHI are reserved for definition by an architecture specific part.

DT_AUXILIARY
Shared object to load before self

DT_FILTER
Shared object to get values from

DT_HIOS
Values from DT_LOOS through DT_HIOS are reserved for definition by specific operating systems.

DT_LOOS
Values from DT_LOOS through DT_HIOS are reserved for definition by specific operating systems.

DT_NUM
Number of dynamic entry tags defined (excepting reserved ranges).

DT_POSFLAG_1
Flags for DT_* entries, effecting the following DT_* entry

DT_RELCOUNT
All Elf32_Rel R_*_RELATIVE relocations have been placed into a single block and this entry specifies the number of entries in that block. This permits ld.so.1 to streamline the processing of RELATIVE relocations.

DT_SYMINENT
Entry size of syminfo

DT_SYMINFO
Address of the Syminfo table.
DT_SYMINsz
    Size of syminfo table (in bytes)

DT_VALRNGHI
    Entries which fall between DT_VALRNGHI & DT_VALRNGLO use the

DT_VALRNGLO
    Entries which fall between DT_VALRNGHI & DT_VALRNGLO use the

DT_VERDEF
    Address of version definition table

DT_VERDEFNUM
    Number of version definitions

DT_VERNEED
    Address of table with needed versions

DT_VERNEEDNUM
    Number of needed versions

DT_VERSYM
    Address of the table provided by the .gnu.version section.
12 C++ Class Representations

12.1 C++ Data Representation

Support for the C++ language shall be as specified in Itanium™ C++ ABI.

Note: This document, although containing a few architecture specific matters, is written as a generic specification, to be usable by C++ implementations on a variety of architectures.

This section provides additional information to supplement Itanium™ C++ ABI. Many of the definitions in that document are made in terms of C++. This section provides additional explanations using C terms to avoid self-referential problems.

12.1.1 Class Representation

An object file generated by the compilation process for a C++ program shall contain several closely related internal objects, or Class Components, to represent each C++ Class. Such objects are not a visible part of the source code. Table 12-1 describes these Class Components at a high level.

<table>
<thead>
<tr>
<th>Object</th>
<th>Contains</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Data</td>
<td>All non-static Class members</td>
</tr>
<tr>
<td>Virtual Table</td>
<td>Information needed to dispatch virtual functions, access virtual base class sub-objects and to access the RTTI information</td>
</tr>
<tr>
<td>RTTI</td>
<td>Run-Time Type Information used by the typeid and dynamic_cast operators, and exception handlers</td>
</tr>
<tr>
<td>Typeinfo Name</td>
<td>String representation of Class name</td>
</tr>
<tr>
<td>Construction Virtual Table</td>
<td>Information needed during construction and destruction of Classes with non-trivial inheritance relationships.</td>
</tr>
<tr>
<td>VTT</td>
<td>A table of virtual table pointers which holds the addresses of construction and non-construction virtual tables.</td>
</tr>
</tbody>
</table>

12.1.1.1 Virtual Table

Virtual tables are specified in Section 2.5.3 of Itanium™ C++ ABI.

Of the various categories of virtual table described in that specification, Category 1 (Leaf) is further described in Figure 12-1 and Category 2 (Non-virtual bases only) is further described in Figure 12-2. LSB conforming systems shall support these categories.

```c
struct {
    ptrdiff_t       baseobject;
    const char      *typeinfo;
    fptr            virtfuncs[0];
};
```

Figure 12-1 Category 1 Virtual Table

```c
struct {
    unsigned long   vcalloffset;
    ptrdiff_t       baseobject;
    const char      *typeinfo;
    fptr            virtfuncs[0];
};
```

Figure 12-2 Category 2 Virtual Table

© 2015 Linux Foundation
Figure 12-2 Category 2 Virtual Table

This specification describes requirements for virtual tables of C++ classes using tables of the following form:

<table>
<thead>
<tr>
<th>Table 12-2 Primary vtable for K (example)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base Offset</strong></td>
</tr>
<tr>
<td><strong>Virtual Base Offset</strong></td>
</tr>
<tr>
<td><strong>RTTI</strong></td>
</tr>
<tr>
<td><strong>vfunc[0]</strong> :</td>
</tr>
<tr>
<td><strong>vfunc[1]</strong> :</td>
</tr>
<tr>
<td><strong>vfunc[2]</strong> :</td>
</tr>
<tr>
<td><strong>vfunc[3]</strong> :</td>
</tr>
<tr>
<td><strong>vfunc[4]</strong> :</td>
</tr>
<tr>
<td><strong>vfunc[5]</strong> :</td>
</tr>
</tbody>
</table>

Each row starting from 'vfunc[i]:' refers to a vtable entry 'vfnc[i]' of a class K, which is an entry for a virtual function A::m, where A is a base class of the class K as described in the *Itanium™ C++ ABI*. This specification requires implementations to interpret the vtable entry information in the following way:

1. A conforming implementation shall contain a vtable of the class K in the specified shared library;
2. The corresponding entry of this vtable 'vfnc[i]' shall be an entry for the virtual function A::m;
3. If the second column of the row contains __cxa_pure_virtual() the corresponding vtable entry of a LSB-conforming implementation shall contain __cxa_pure_virtual() or 'Y::m', where Y is the class K, the class A or a base class of the class K derived from the class A.

   **Note:** In this case virtual function A::m in class K is considered to be specified as pure virtual by this specification.

4. If the second column of the row contains 'X::m' the corresponding vtable entry of a LSB-conforming implementation shall contain 'Y::m', where Y is the class K, the class X or a base class of the class K derived from the class X.
5. If the second column of the row contains 'NULL or X::m' the corresponding vtable entry of a LSB-conforming implementation shall contain NULL or 'Y::m', where Y is the class K, the class X or a base class of the class K derived from the class X.

   **Note:** In this case virtual function A::m in class K is considered to be specified as inline by this specification.

An application may use any non-pure virtual function specified in this specification, and can expect the specified behavior irrespective of which particular method implements this functionality. An application may not use inline virtual functions at the binary level since its vtable entry may be NULL.

### 12.1.1.2 Run-Time Type Information

Each type used in a C++ program has a data structure associated with it that provide information about the type which is used at runtime. This Run Time Type Information (RTTI) is defined in section 2.9.5 in *Itanium™ C++ ABI*. Additional details about the layout of this data is provided here.
struct {
    void      *basevtable;
    char      *name;
};

Figure 12-3 Run-Time Type Information Prefix

struct {
    void      *basevtable;
    char      *name;
    void      *basetypeinfo[0];
};

Figure 12-4 Run-Time Type Information For Classes with no base class

struct {
    void      *basevtable;
    char      *name;
    void      *basetype;
    void      *basetypeinfo[0];
};

Figure 12-5 Run-Time Type Information for Classes with a single base class

struct base_type_info {
    char    *base_type;
    unsigned long   offset_flags;
};

struct {
    void      *basevtable;
    char      *name;
    unsigned int    flags;
    unsigned int    base_count;
    struct base_type_info base_info[0];
};

Figure 12-6 Run-Time Type Information for classes with multiple inheritance

struct {
    void      *basevtable;
    char      *name;
    unsigned int    flags;
    void      *pointee;
    void      *basetypeinfo[0];
};

Figure 12-7 Run-Time Type Information for pointer types

struct {
    void      *basevtable;
    char      *name;
    unsigned int    flags;
    void      *pointee;
    void      *context;
    void      *basetypeinfo[0];
};

Figure 12-8 Run-Time Type Information for pointer to member types
13 Symbol Mapping

This chapter defines how names are mapped from the source symbol to the object symbol.

13.1 Symbol Mapping

Symbols in a source program are translated by the compilation system into symbols that exist in the object file. The rules for this translation are defined here.

13.1.1 C++ Language

External symbol names in a C++ object file shall be encoded according to the "name mangling" rules described in the Itanium™ C++ ABI.
III Base Libraries
14 Base Libraries

14.1 Introduction

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

- libc
- libm
- libgcc_s
- libdl
- librt
- libcrypt
- libpam

There are three main parts to the definition of each of these libraries.

The "Interfaces" section defines the required library name and version, and the required public symbols (interfaces and global data), as well as symbol versions, if any.

The "Interface Definitions" section provides complete or partial definitions of certain interfaces where either this specification is the source specification, or where there are variations from the source specification. If an interface definition requires one or more header files, one of those headers shall include the function prototype for the interface.

For source definitions of interfaces which include a reference to a header file, the contents of such header files form a part of the specification. The "Data Definitions" section provides the binary-level details for the header files from the source specifications, such as values for macros and enumerated types, as well as structure layouts, sizes and padding, etc. These data definitions, although presented in the form of header files for convenience, should not be taken a representing complete header files, as they are a supplement to the source specifications. Application developers should follow the guidelines of the source specifications when determining which header files need to be included to completely resolve all references.

**Note:** While the Data Definitions supplement the source specifications, this specification itself does not require conforming implementations to supply any header files.

14.2 Program Interpreter

The Program Interpreter is specified in the appropriate architecture specific part of the LSB Core Specification.

14.3 Interfaces for libc

Table 14-1 defines the library name and shared object name for the libc library

<table>
<thead>
<tr>
<th>Library:</th>
<th>libc</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>See architecture specific part.</td>
</tr>
</tbody>
</table>

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

[LFS] Large File Support
[LSB] This Specification
[RPC + XDR] RFC 5531/4506 RPC & XDR
[SUSv2] SUSv2
14.3.1 RPC

14.3.1.1 Interfaces for RPC

An LSB conforming implementation shall provide the generic functions for RPC specified in Table 14-2, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Compatibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>authnone_create</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>callrpc</td>
<td>[RPC + XDR]</td>
</tr>
<tr>
<td>clnt_create</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>clnt_pccreateerror</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>clnt_permrno</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>clnt_pererror</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>clnt_spcreateerror</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>clnt_spermrno</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>clntraw_create</td>
<td>[RPC + XDR]</td>
</tr>
<tr>
<td>clnttcp_create</td>
<td>[RPC + XDR]</td>
</tr>
<tr>
<td>clntudp_create</td>
<td>[RPC + XDR]</td>
</tr>
<tr>
<td>clntudp_bufcreate</td>
<td>[RPC + XDR]</td>
</tr>
<tr>
<td>clnt_perror</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>clnt_sperrno</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>clnt_sperror</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>clnt_spcreateerror</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>clntsperror</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>key_decryptsession</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>map_getport</td>
<td>[LSB]</td>
</tr>
<tr>
<td>map_set</td>
<td>[LSB]</td>
</tr>
<tr>
<td>svc_getreqset</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>svc_register</td>
<td>[LSB]</td>
</tr>
<tr>
<td>svc_run</td>
<td>[LSB]</td>
</tr>
<tr>
<td>svc_sendreply</td>
<td>[LSB]</td>
</tr>
<tr>
<td>svcerr_auth</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>svcerr_decode</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>svcerr_noproc</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>svcerr_noprogs</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>svcerr_systemerr</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>svcerr_weakauth</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>svcfd_create</td>
<td>[RPC + XDR]</td>
</tr>
<tr>
<td>svcre_ar_create</td>
<td>[RPC + XDR]</td>
</tr>
<tr>
<td>svctcp_create</td>
<td>[LSB]</td>
</tr>
<tr>
<td>svcudp_create</td>
<td>[LSB]</td>
</tr>
<tr>
<td>xdr_accepted_reply</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_array</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_bool</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_bytes</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_callhdr</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_callmsg</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_char</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_double</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_enum</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_float</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_free</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_int</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_long</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_opaque</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_opaque_auth</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_pointer</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_reference</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_rejected_reply</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_replmsg</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_short</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_string</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_u_char</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_u_int</td>
<td>[LSB]</td>
</tr>
<tr>
<td>xdr_u_long</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_u_short</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_union</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_vector</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdr_void</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdrwrapstring</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdmem_create</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdrrec_create</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdrrec_endofrecord</td>
<td>[RPC + XDR]</td>
</tr>
<tr>
<td>xdrrec_eof</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdrrec_skiprecording</td>
<td>[SVID.4]</td>
</tr>
<tr>
<td>xdrstdio_create</td>
<td>[LSB]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic deprecated functions for RPC specified in Table 14-3, 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.
14.3.2 Epoll

14.3.2.1 Interfaces for Epoll

An LSB conforming implementation shall provide the generic functions for Epoll specified in Table 14-4, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-4 libc - Epoll Function Interfaces

| epoll_create(GLIBC_2.3.2) [LSB] | epoll_ctl(GLIBC_2.3.2) [LSB] | epoll_wait(GLIBC_2.3.2) [LSB] |

14.3.3 System Calls

14.3.3.1 Interfaces for System Calls

An LSB conforming implementation shall provide the generic functions for System Calls specified in Table 14-5, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-5 libc - System Calls Function Interfaces

<p>| __chk_fail(GLIBC_2.3.4) [LSB] | __fxstat [LSB] | __fstatat(GLIBC_2.4) [LSB] | __getgroups_chk(GLIBC_2.4) [LSB] |
| __getgid [LSB] | __lxstat [LSB] | __read_chk(GLIBC_2.4) [LSB] | __readlink_chk(GLIBC_2.4) [LSB] |
| __stack_chk_fail(GLIBC_2.4) [LSB] | __xmknod [LSB] | __xmknodat(GLIBC_2.4) [LSB] | __xstat [LSB] |
| faccessat(GLIBC_2.4) [SUSv4] | fchdir [SUSv4] | fchmod [SUSv4] | fchmodat(GLIBC_2.4) [SUSv4] |
| fchown [SUSv4] | fchownat(GLIBC_2.4) [SUSv4] | fcntl [LSB] | fdatasync [SUSv4] |
| ftruncate [SUSv4] | futimens(GLIBC_2.6) [SUSv4] | futimes(GLIBC_2.3) [LSB] | getcontext [SUSv3] |</p>
<table>
<thead>
<tr>
<th>Function</th>
<th>LSB Core - Generic 5.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>getpriority [SUSv4]</td>
<td>getpriority [SUSv4]</td>
</tr>
<tr>
<td>getuid [SUSv4]</td>
<td>getwd [SUSv3]</td>
</tr>
<tr>
<td>kill [LSB]</td>
<td>killpg [SUSv4]</td>
</tr>
<tr>
<td>linkat(GLIBC_2.4) [SUSv4]</td>
<td>lchown [SUSv4]</td>
</tr>
<tr>
<td>mkdir [SUSv4]</td>
<td>mkdirat(GLIBC_2.4) [SUSv4]</td>
</tr>
<tr>
<td>mlock [SUSv4]</td>
<td>munlock [SUSv4]</td>
</tr>
<tr>
<td>munmap [SUSv4]</td>
<td>munlockall [SUSv4]</td>
</tr>
<tr>
<td>nice [SUSv4]</td>
<td>open [SUSv4]</td>
</tr>
<tr>
<td>opendir [SUSv4]</td>
<td>pathconf [SUSv4]</td>
</tr>
<tr>
<td>poll [SUSv4]</td>
<td>pread [SUSv4]</td>
</tr>
<tr>
<td>ptrace [LSB]</td>
<td>read [SUSv4]</td>
</tr>
<tr>
<td>readdir [SUSv4]</td>
<td>readdir [SUSv4]</td>
</tr>
<tr>
<td>rename [SUSv4]</td>
<td>rmdir [SUSv4]</td>
</tr>
<tr>
<td>sched_get_priority_max [SUSv4]</td>
<td>sched_get_priority_min [SUSv4]</td>
</tr>
<tr>
<td>sched_getaffinity(GLIBC_2.3.4) [LSB]</td>
<td>sched_setparam [SUSv4]</td>
</tr>
<tr>
<td>sched_yield [SUSv4]</td>
<td>select [SUSv4]</td>
</tr>
<tr>
<td>setegid [SUSv4]</td>
<td>seteuid [SUSv4]</td>
</tr>
<tr>
<td>setgid [SUSv4]</td>
<td>setgid [SUSv4]</td>
</tr>
<tr>
<td>setpriority [SUSv4]</td>
<td>setsid [SUSv4]</td>
</tr>
<tr>
<td>setreuid [SUSv4]</td>
<td>setregid [SUSv4]</td>
</tr>
<tr>
<td>setrlimit [LSB]</td>
<td>setrlimit64 [LFS]</td>
</tr>
<tr>
<td>setrlimit [LSB]</td>
<td>setrlimit64 [LSB]</td>
</tr>
<tr>
<td>sleep [SUSv4]</td>
<td>statfs [SUSv4]</td>
</tr>
<tr>
<td>symlink [LSB]</td>
<td>symlink [SUSv4]</td>
</tr>
<tr>
<td>symlink(GLIBC_2.4) [SUSv4]</td>
<td>symlink(GLIBC_2.4) [SUSv4]</td>
</tr>
<tr>
<td>sysconf [LSB]</td>
<td>sysinfo [LSB]</td>
</tr>
<tr>
<td>ulimit [SUSv4]</td>
<td>umask [SUSv4]</td>
</tr>
<tr>
<td>utimes [SUSv4]</td>
<td>utimensat(GLIBC_2.6) [SUSv4]</td>
</tr>
<tr>
<td>vfork [SUSv3]</td>
<td>wait [SUSv4]</td>
</tr>
<tr>
<td>waitid [SUSv4]</td>
<td>wait4 [LSB]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic deprecated functions for System Calls specified in Table 14-6, 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.

### 14.3.4 Standard I/O

#### 14.3.4.1 Interfaces for Standard I/O

An LSB conforming implementation shall provide the generic functions for Standard I/O specified in [Table 14-7](#), with the full mandatory functionality as described in the referenced underlying specification.

---

<table>
<thead>
<tr>
<th>Table 14-7 libc - System Calls Deprecated Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fstatfs</code></td>
</tr>
<tr>
<td>[LSB]</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Table 14-7 libc - Standard I/O Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_IO_feof</code></td>
</tr>
<tr>
<td>[LSB]</td>
</tr>
<tr>
<td><code>__fgets_chk(GLIBC_2.4)</code></td>
</tr>
<tr>
<td>[LSB]</td>
</tr>
<tr>
<td><code>__printf_chk</code></td>
</tr>
<tr>
<td>[LSB]</td>
</tr>
<tr>
<td><code>__vprintf_chk</code></td>
</tr>
<tr>
<td>[LSB]</td>
</tr>
<tr>
<td><code>clearerr</code></td>
</tr>
<tr>
<td>[SUSv4]</td>
</tr>
<tr>
<td><code>fclose</code></td>
</tr>
<tr>
<td>[SUSv4]</td>
</tr>
<tr>
<td><code>ferror</code></td>
</tr>
<tr>
<td>[SUSv4]</td>
</tr>
<tr>
<td><code>fgetc</code></td>
</tr>
<tr>
<td>[SUSv4]</td>
</tr>
<tr>
<td><code>fgets_unlocked</code></td>
</tr>
<tr>
<td>[LSB]</td>
</tr>
<tr>
<td><code>fgetws_unlocked</code></td>
</tr>
<tr>
<td>[LSB]</td>
</tr>
<tr>
<td><code>fileno</code></td>
</tr>
<tr>
<td>[SUSv4]</td>
</tr>
<tr>
<td><code>fputc</code></td>
</tr>
<tr>
<td>[SUSv4]</td>
</tr>
<tr>
<td><code>fputwc_unlocked</code></td>
</tr>
<tr>
<td>[LSB]</td>
</tr>
<tr>
<td><code>freopen</code></td>
</tr>
<tr>
<td>[SUSv4]</td>
</tr>
<tr>
<td><code>fsetpos</code></td>
</tr>
<tr>
<td>[SUSv4]</td>
</tr>
<tr>
<td><code>fwrite_unlocked</code></td>
</tr>
<tr>
<td>[LSB]</td>
</tr>
<tr>
<td><code>getchar_unlocked</code></td>
</tr>
<tr>
<td>[SUSv4]</td>
</tr>
<tr>
<td><code>getwc_unlocked</code></td>
</tr>
<tr>
<td>[LSB]</td>
</tr>
<tr>
<td><code>printf</code></td>
</tr>
<tr>
<td>[SUSv4]</td>
</tr>
<tr>
<td><code>putchar_unlocked</code></td>
</tr>
<tr>
<td>[SUSv4]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic deprecated functions for Standard I/O specified in Table 14-8, 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 14-8 libc - Standard I/O Deprecated Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>tempnam</td>
<td>[SUSv4]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Standard I/O specified in Table 14-9, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-9 libc - Standard I/O Data Interfaces

<table>
<thead>
<tr>
<th>Variable</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>stderr</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>stdin</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>stdout</td>
<td>[SUSv4]</td>
</tr>
</tbody>
</table>

14.3.5 Signal Handling

14.3.5.1 Interfaces for Signal Handling

An LSB conforming implementation shall provide the generic functions for Signal Handling specified in Table 14-10, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-10 libc - Signal Handling Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>__libc_current_sigrtmin</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__libc_current_sigrtmax</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__sigsetjmp</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__sysv_signal</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__xpg_sigpause</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__bsd_signal</td>
<td>[SUSv3]</td>
</tr>
<tr>
<td>psiginfo(GLIBC_2.10)</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>psignal</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigaction</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigaddset</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigaltstack</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigandset</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigdelset</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigemptyset</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigfillset</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>siginterrupt</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigisemptyset</td>
<td>[LSB]</td>
</tr>
<tr>
<td>sigismember</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>siglongjmp</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>signal</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigorset</td>
<td>[LSB]</td>
</tr>
<tr>
<td>sigpending</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigprocmask</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigqueue</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigreturn</td>
<td>[LSB]</td>
</tr>
<tr>
<td>sigset</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigsuspend</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigtimedwait</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigwait</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sigwaitinfo</td>
<td>[SUSv4]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic deprecated functions for
Signal Handling specified in Table 14-11, 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 14-11 libc - Signal Handling Deprecated Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>sigpause</td>
<td>[LSB]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Signal Handling specified in Table 14-12, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-12 libc - Signal Handling Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>_sys_siglist</td>
<td>[LSB]</td>
</tr>
</tbody>
</table>

### 14.3.6 Localization Functions

#### 14.3.6.1 Interfaces for Localization Functions

An LSB conforming implementation shall provide the generic functions for Localization Functions specified in Table 14-13, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-13 libc - Localization Functions Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>LSB</th>
<th>SUSv4</th>
</tr>
</thead>
<tbody>
<tr>
<td>bind_textdomain_codeset</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>bindtextdomain</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>catclose</td>
<td>[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>catgets</td>
<td>[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>catopen</td>
<td>[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>dcmgetext</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>dcngetext</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>dgettext</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>dcngettext</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>dgettext</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>duplocale(GLIBC_2.3)</td>
<td>[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>freelocale(GLIBC_2.3)</td>
<td>[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>gettext</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>localeconv</td>
<td>[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>newlocale(GLIBC_2.3)</td>
<td>[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>ngettext</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>nl_langinfo</td>
<td>[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>setlocale</td>
<td>[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>textdomain</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>uselocale(GLIBC_2.3)</td>
<td>[SUSv4]</td>
<td></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Localization Functions specified in Table 14-14, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-14 libc - Localization Functions Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>_nl_msg_cat_cntr</td>
<td>[LSB]</td>
</tr>
</tbody>
</table>

### 14.3.7 Posix Spawn Option

#### 14.3.7.1 Interfaces for Posix Spawn Option

An LSB conforming implementation shall provide the generic functions for Posix Spawn Option specified in Table 14-15, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-15 libc - Posix Spawn Option Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>SUSv4</th>
<th>SUSv4</th>
<th>SUSv4</th>
<th>SUSv4</th>
</tr>
</thead>
<tbody>
<tr>
<td>posix_spawn</td>
<td>[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>posix_spawn_file_actions_addclose</td>
<td>[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>posix_spawn_file_actions_adddup2</td>
<td>[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>posix_spawn_file_actions_addopen</td>
<td>[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 14.3.8 Posix Advisory Option

#### 14.3.8.1 Interfaces for Posix Advisory Option

An LSB conforming implementation shall provide the generic functions for Posix Advisory Option specified in **Table 14-16**, with the full mandatory functionality as described in the referenced underlying specification.

**Table 14-16 libc - Posix Advisory Option Function Interfaces**

<table>
<thead>
<tr>
<th>Function</th>
<th>Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>posix_fadvise</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>posix_fallocate</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>posix_madvise</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>posix_memalign</td>
<td>[SUSv4]</td>
</tr>
</tbody>
</table>

### 14.3.9 Socket Interface

#### 14.3.9.1 Interfaces for Socket Interface

An LSB conforming implementation shall provide the generic functions for Socket Interface specified in **Table 14-17**, with the full mandatory functionality as described in the referenced underlying specification.

**Table 14-17 libc - Socket Interface Function Interfaces**

<table>
<thead>
<tr>
<th>Function</th>
<th>Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>__gethostname_chk(GLIBC_2.4)</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__h_errno_locatio</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__recvchk(GLIBC_C_2.4)</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__recvfromchk(GLIBC_2.4)</td>
<td>[LSB]</td>
</tr>
<tr>
<td>accept</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>bind</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>bindresvport</td>
<td>[LSB]</td>
</tr>
<tr>
<td>connect</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>gethostid</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>gethostname</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>getifaddrs(GLIBC_2.3)</td>
<td>[LSB]</td>
</tr>
<tr>
<td>getpeername</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>getsockname</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>getsockopt</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>if_freenameindex</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>if_indextoname</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>if_nametoindex</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>if_nameindex</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>listen</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>recv</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>recvfrom</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>recvmsg</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>send</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sendto</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>setsockopt</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>shutdown</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>socket</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>socketpair</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>sockatmark</td>
<td>[SUSv4]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Socket Interface specified in Table 14-18, with the full mandatory functionality as described in the referenced underlying specification.

**Table 14-18 libc - Socket Interface Data Interfaces**

<table>
<thead>
<tr>
<th>in6addr_any</th>
<th>in6addr_loopback</th>
</tr>
</thead>
<tbody>
<tr>
<td>[SUSv3]</td>
<td>[SUSv3]</td>
</tr>
</tbody>
</table>

### 14.3.10 Wide Characters

#### 14.3.10.1 Interfaces for Wide Characters

An LSB conforming implementation shall provide the generic functions for Wide Characters specified in Table 14-19, with the full mandatory functionality as described in the referenced underlying specification.

**Table 14-19 libc - Wide Characters Function Interfaces**

<table>
<thead>
<tr>
<th>__fgetws_chk(GLIBC_2.4) [LSB]</th>
<th>__fwprintf_chk(GLIBC_2.4) [LSB]</th>
<th>__mbsntowcs_chk(GLIBC_2.4) [LSB]</th>
<th>__mbstowcs_chk(GLIBC_2.4) [LSB]</th>
</tr>
</thead>
<tbody>
<tr>
<td>__fwscanf_chk(GLIBC_2.4) [LSB]</td>
<td>__fwscanf_chk(GLIBC_2.4) [LSB]</td>
<td>__vfwprintf_chk(GLIBC_2.4) [LSB]</td>
<td>__vfwscanf_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcpcpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wcpcpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wmemcpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wmemcpy_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcpncpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wcpncpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wmempcpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wmempcpy_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcrtomb_chk(GLIBC_2.4) [LSB]</td>
<td>__wcrtomb_chk(GLIBC_2.4) [LSB]</td>
<td>__wmemset_chk(GLIBC_2.4) [LSB]</td>
<td>__wmemset_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcscat_chk(GLIBC_2.4) [LSB]</td>
<td>__wcscat_chk(GLIBC_2.4) [LSB]</td>
<td>__wprintf_chk(GLIBC_2.4) [LSB]</td>
<td>__wprintf_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcscpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wcscpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wmemcpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wmemcpy_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcsncat_chk(GLIBC_2.4) [LSB]</td>
<td>__wcsncat_chk(GLIBC_2.4) [LSB]</td>
<td>__wmempcpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wmempcpy_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcsncpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wcsncpy_chk(GLIBC_2.4) [LSB]</td>
<td>__wcstombs_chk(GLIBC_2.4) [LSB]</td>
<td>__wcstombs_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcstol_internal [LSB]</td>
<td>__wcstold_internal [LSB]</td>
<td>__wctomb_chk(GLIBC_2.4) [LSB]</td>
<td>__wctomb_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcsrtombs_chk(GLIBC_2.4) [LSB]</td>
<td>__wcsrtombs_chk(GLIBC_2.4) [LSB]</td>
<td>__wcstod_internal [LSB]</td>
<td>__wcstod_internal [LSB]</td>
</tr>
<tr>
<td>__wctol_internal [LSB]</td>
<td>__wctol_internal [LSB]</td>
<td>__wctomb_chk(GLIBC_2.4) [LSB]</td>
<td>__wctomb_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcscat(GLIBC_2.4) [LSB]</td>
<td>__wcscat(GLIBC_2.4) [LSB]</td>
<td>__wprintf(GLIBC_2.4) [LSB]</td>
<td>__wprintf(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcscpy(GLIBC_2.4) [LSB]</td>
<td>__wcscpy(GLIBC_2.4) [LSB]</td>
<td>__wmemcpy(GLIBC_2.4) [LSB]</td>
<td>__wmemcpy(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcsncat(GLIBC_2.4) [LSB]</td>
<td>__wcsncat(GLIBC_2.4) [LSB]</td>
<td>__wmempcpy(GLIBC_2.4) [LSB]</td>
<td>__wmempcpy(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcsncpy(GLIBC_2.4) [LSB]</td>
<td>__wcsncpy(GLIBC_2.4) [LSB]</td>
<td>__wcstombs(GLIBC_2.4) [LSB]</td>
<td>__wcstombs(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcstol(GLIBC_2.4) [LSB]</td>
<td>__wcstold(GLIBC_2.4) [LSB]</td>
<td>__wctomb(GLIBC_2.4) [LSB]</td>
<td>__wctomb(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcsrtombs(GLIBC_2.4) [LSB]</td>
<td>__wcsrtombs(GLIBC_2.4) [LSB]</td>
<td>__wcstod(GLIBC_2.4) [LSB]</td>
<td>__wcstod(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wctol(GLIBC_2.4) [LSB]</td>
<td>__wctol(GLIBC_2.4) [LSB]</td>
<td>__wctomb(GLIBC_2.4) [LSB]</td>
<td>__wctomb(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__wcscat([SUSv4])</td>
<td>__wcscat([SUSv4])</td>
<td>__wprintf([SUSv4])</td>
<td>__wprintf([SUSv4])</td>
</tr>
<tr>
<td>__wcscpy([SUSv4])</td>
<td>__wcscpy([SUSv4])</td>
<td>__wmemcpy([SUSv4])</td>
<td>__wmemcpy([SUSv4])</td>
</tr>
<tr>
<td>__wcsncat([SUSv4])</td>
<td>__wcsncat([SUSv4])</td>
<td>__wmempcpy([SUSv4])</td>
<td>__wmempcpy([SUSv4])</td>
</tr>
<tr>
<td>__wcsncpy([SUSv4])</td>
<td>__wcsncpy([SUSv4])</td>
<td>__wcstombs([SUSv4])</td>
<td>__wcstombs([SUSv4])</td>
</tr>
<tr>
<td>__wcstol([SUSv4])</td>
<td>__wcstold([SUSv4])</td>
<td>__wctomb([SUSv4])</td>
<td>__wctomb([SUSv4])</td>
</tr>
<tr>
<td>__wcsrtombs([SUSv4])</td>
<td>__wcsrtombs([SUSv4])</td>
<td>__wcstod([SUSv4])</td>
<td>__wcstod([SUSv4])</td>
</tr>
<tr>
<td>__wctol([SUSv4])</td>
<td>__wctol([SUSv4])</td>
<td>__wctomb([SUSv4])</td>
<td>__wctomb([SUSv4])</td>
</tr>
<tr>
<td>fgetws([SUSv4])</td>
<td>fputwc([SUSv4])</td>
<td>fputws([SUSv4])</td>
<td>fwide([SUSv4])</td>
</tr>
<tr>
<td>fprintf([SUSv4])</td>
<td>fwscanf([SUSv4])</td>
<td>getwc([SUSv4])</td>
<td>getwchar([SUSv4])</td>
</tr>
<tr>
<td>iswalnum_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswalnum_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswalpha_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswalpha_l(GLIBC_C_2.3) [SUSv4]</td>
</tr>
<tr>
<td>iswblank_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswblank_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswcntrl_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswcntrl_l(GLIBC_C_2.3) [SUSv4]</td>
</tr>
<tr>
<td>iswctype_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswctype_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswdigit_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswdigit_l(GLIBC_C_2.3) [SUSv4]</td>
</tr>
<tr>
<td>iswgraph_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswgraph_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswlower_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswlower_l(GLIBC_C_2.3) [SUSv4]</td>
</tr>
<tr>
<td>iswprint_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswprint_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswpunct_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswpunct_l(GLIBC_C_2.3) [SUSv4]</td>
</tr>
<tr>
<td>iswspace_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswspace_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswupper_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswupper_l(GLIBC_C_2.3) [SUSv4]</td>
</tr>
<tr>
<td>iswxdigit_l(GLIBC_C_2.3) [SUSv4]</td>
<td>iswxdigit_l(GLIBC_C_2.3) [SUSv4]</td>
<td>mbrelen([SUSv4])</td>
<td>mbrelen([SUSv4])</td>
</tr>
<tr>
<td>mbsinit([SUSv4])</td>
<td>mbsinit([SUSv4])</td>
<td>mbtowc([SUSv4])</td>
<td>mbtowc([SUSv4])</td>
</tr>
<tr>
<td>mbsrtowcs([SUSv4])</td>
<td>mbsrtowcs([SUSv4])</td>
<td>mbstowcs([SUSv4])</td>
<td>mbstowcs([SUSv4])</td>
</tr>
<tr>
<td>mbtowc([SUSv4])</td>
<td>mbtowc([SUSv4])</td>
<td>putwchar([SUSv4])</td>
<td>putwchar([SUSv4])</td>
</tr>
<tr>
<td>putwc([SUSv4])</td>
<td>putwc([SUSv4])</td>
<td>swprintf([SUSv4])</td>
<td>swprintf([SUSv4])</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
**14.3.11 String Functions**

**14.3.11.1 Interfaces for String Functions**

An LSB conforming implementation shall provide the generic functions for String Functions specified in Table 14-20, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>__memcpy_chk(GLIBC_2.3.4)</td>
<td>Copy memory with type checking</td>
</tr>
<tr>
<td>__memmove_chk(GLIBC_2.3.4)</td>
<td>Move memory with type checking</td>
</tr>
<tr>
<td>__memset_chk(GLIBC_2.3.4)</td>
<td>Null-terminate memory with type checking</td>
</tr>
</tbody>
</table>
| __rawmemchr | Retrieve character without_final {}
| __strcat_chk(GLIBC_2.3.4) | Concatenate two strings with type checking |
| __strcpy_chk(GLIBC_2.3.4) | Copy string with type checking |
| __strdup [LSB] | Duplicate string |
| __stpcpy [LSB] | Retrieve end of string without_final {}
| __stpcpy_chk(GLIBC_2.3.4) | Retrieve end of string with type checking |
| __strncat_chk(GLIBC_2.3.4) | Concatenate two strings with type checking |
| __strncpy_chk(GLIBC_2.3.4) | Copy string with type checking |
| __strncpy [LSB] | Copy string |
| __strcspn [LSB] | Retrieve end of string without_final {}
| __strchr [LSB] | Retrieve character |
| __strupr [LSB] | Transform string to uppercase |
| __strtok [LSB] | Retrieve token |
| __strtokr [LSB] | Retrieve token with type checking |
| __strchr [LSB] | Retrieve character |
| __strncpy [LSB] | Copy string |
| __strncpy_chk(GLIBC_2.3.4) | Copy string with type checking |
| __strcspn [LSB] | Retrieve end of string without_final {}
| __strchr [LSB] | Retrieve character |
| __strncpy [LSB] | Copy string |
| __strncpy_chk(GLIBC_2.3.4) | Copy string with type checking |
| __strcspn [LSB] | Retrieve end of string without_final {}
| __strchr [LSB] | Retrieve character |
| __strncpy [LSB] | Copy string |
| __strncpy_chk(GLIBC_2.3.4) | Copy string with type checking |
| __strcspn [LSB] | Retrieve end of string without_final {}
An LSB conforming implementation shall provide the generic deprecated functions for String Functions specified in Table 14-21, 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 14-21 libc - String Functions Deprecated Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>strerror_r [LSB]</td>
<td>strerror_r [LSB]</td>
</tr>
</tbody>
</table>

### 14.3.12 IPC Functions

#### 14.3.12.1 Interfaces for IPC Functions

An LSB conforming implementation shall provide the generic functions for IPC Functions specified in Table 14-22, with the full mandatory functionality as described in the referenced underlying specification.

### Table 14-22 libc - IPC Functions Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ftok [SUSv4]</td>
<td>ftok [SUSv4]</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
14 Base Libraries

|----------------|----------------|----------------|---------------|

14.3.13 Regular Expressions

14.3.13.1 Interfaces for Regular Expressions

An LSB conforming implementation shall provide the generic functions for Regular Expressions specified in Table 14-23, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-23 libc - Regular Expressions Function Interfaces


14.3.14 Character Type Functions

14.3.14.1 Interfaces for Character Type Functions

An LSB conforming implementation shall provide the generic functions for Character Type Functions specified in Table 14-24, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-24 libc - Character Type Functions Function Interfaces

<table>
<thead>
<tr>
<th>_ctype_b_loc(GLIBC_2.3) [LSB]</th>
<th>_ctype_get_mb_cur_max [LSB]</th>
<th>_ctype_tolower_loc(GLIBC_2.3) [LSB]</th>
<th>_ctype_toupper_loc(GLIBC_2.3) [LSB]</th>
</tr>
</thead>
</table>

14.3.15 Time Manipulation

14.3.15.1 Interfaces for Time Manipulation

An LSB conforming implementation shall provide the generic functions for Time Manipulation specified in Table 14-25, with the full mandatory functionality as described in the referenced underlying specification.
An LSB conforming implementation shall provide the generic data interfaces for Time Manipulation specified in Table 14-26, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-26 libc - Time Manipulation Data Interfaces

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>__timezone [LSB]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__tzname [LSB]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

14.3.16 Terminal Interface Functions

14.3.16.1 Interfaces for Terminal Interface Functions

An LSB conforming implementation shall provide the generic functions for Terminal Interface Functions specified in Table 14-27, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-27 libc - Terminal Interface Functions Function Interfaces

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>tcsendbreak [SUSv4]</td>
<td>tcsetattr [SUSv4]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

14.3.17 System Database Interface

14.3.17.1 Interfaces for System Database Interface

An LSB conforming implementation shall provide the generic functions for System Database Interface specified in Table 14-28, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-28 libc - System Database Interface Function Interfaces

|------------------|---------------------|-----------------|--------------------|
An LSB conforming implementation shall provide the generic deprecated functions for System Database Interface specified in Table 14-29, 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 14-29 libc - System Database Interface Deprecated Function Interfaces

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>gethostbyname2_r [LSB]</td>
<td>gethostbyname_r [LSB]</td>
<td>gethostbyname2 [LSB]</td>
<td>gethostbyname2_r [LSB]</td>
</tr>
</tbody>
</table>

14.3.18 Language Support

14.3.18.1 Interfaces for Language Support

An LSB conforming implementation shall provide the generic functions for Language Support specified in Table 14-30, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-30 libc - Language Support Function Interfaces

| __libc_start_main [LSB] | __register_atfork(GLIBC_2.3.2) [LSB] |

14.3.19 Large File Support

14.3.19.1 Interfaces for Large File Support

An LSB conforming implementation shall provide the generic functions for Large File Support specified in Table 14-31, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-31 libc - Large File Support Function Interfaces

<table>
<thead>
<tr>
<th>__fxstat64 [LSB]</th>
<th>__fxstatat64(GLIBC_2.4) [LSB]</th>
<th>__lxstat64 [LSB]</th>
<th>__xstat64 [LSB]</th>
</tr>
</thead>
<tbody>
<tr>
<td>creat64 [LFS]</td>
<td>fgetpos64 [LFS]</td>
<td>fopen64 [LFS]</td>
<td>freopen64 [LFS]</td>
</tr>
<tr>
<td>fseeko64 [LFS]</td>
<td>fsetpos64 [LFS]</td>
<td>fstat64 [LSB]</td>
<td>fstatvfs64 [LFS]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic deprecated functions for Large File Support specified in Table 14-32, 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>
<thead>
<tr>
<th>Table 14-32 libc - Large File Support Deprecated Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>fstatfs64 [LSB]</td>
</tr>
</tbody>
</table>

### 14.3.20 Inotify

#### 14.3.20.1 Interfaces for Inotify

An LSB conforming implementation shall provide the generic functions for Inotify specified in Table 14-33, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 14-33 libc - Inotify Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>inotify_add_watch(GLIBC_2.4) [LSB]</td>
</tr>
</tbody>
</table>

### 14.3.21 Standard Library

#### 14.3.21.1 Interfaces for Standard Library

An LSB conforming implementation shall provide the generic functions for Standard Library specified in Table 14-34, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 14-34 libc - Standard Library Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>_Exit [SUSv4]</td>
</tr>
<tr>
<td>__cxa_finalize [LSB]</td>
</tr>
<tr>
<td>__getlogin_r_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>_pread64_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>__syslog_chk(GLIBC_2.4) [LSB]</td>
</tr>
<tr>
<td>Function</td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>drand48_r</td>
</tr>
<tr>
<td>envz_get</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic deprecated functions for Standard Library specified in Table 14-35, 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 14-35 libc - Standard Library Deprecated Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>basename</td>
<td>[LSB]</td>
</tr>
<tr>
<td>getdomainname</td>
<td>[LSB]</td>
</tr>
<tr>
<td>inet_aton</td>
<td>[LSB]</td>
</tr>
<tr>
<td>tmpnam</td>
<td>[SUSv4]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Standard Library specified in Table 14-36, with the full mandatory functionality as described in the referenced underlying specification.

### Table 14-36 libc - Standard Library Data Interfaces

<table>
<thead>
<tr>
<th>Variable</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>__environ</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__sys_errlist</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__getdate_err</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>optarg</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>opterr</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>optind</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>__environ</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>__environ</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>__sys_errlist</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>__getdate_err</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>optarg</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>opterr</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>optind</td>
<td>[SUSv4]</td>
</tr>
</tbody>
</table>

### 14.3.22 GNU Extensions for libc

#### 14.3.22.1 Interfaces for GNU Extensions for libc

An LSB conforming implementation shall provide the generic functions for GNU Extensions for libc specified in Table 14-37, with the full mandatory functionality as described in the referenced underlying specification.

### Table 14-37 libc - GNU Extensions for libc Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>gnu_get_libc_release</td>
<td>[LSB]</td>
</tr>
<tr>
<td>gnu_get_libc_version</td>
<td>[LSB]</td>
</tr>
</tbody>
</table>
14.4 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.

14.4.1 argz.h

typedef int error_t;
extern error_t argz_add(char **argz, size_t * argz_len, const char *str);
extern error_t argz_add_sep(char **argz, size_t * argz_len, const char *str, int sep);
extern error_t argz_append(char **argz, size_t * argz_len, const char *buf, size_t buf_len);
extern size_t argz_count(const char *argz, size_t * argz_len);
extern error_t argz_create(char *const argv[], char **argz, size_t * argz_len);
extern error_t argz_create_sep(const char *str, int sep, char **argz, size_t * argz_len);
extern void argz_delete(char **argz, size_t * argz_len, char *entry);
extern void argz_extract(const char *argz, size_t argz_len, char **argv);
extern error_t argz_insert(char **argz_insert, size_t * argz_len, char *before, const char *entry);
extern char argz_next(const char *argz, size_t argz_len, const char *entry);
extern error_t argz_replace(char **argz, size_t * argz_len, const char *str, const char *with, unsigned int *replace_count);
extern void argz_stringify(char *argz, size_t argz_len, int sep);

14.4.2 arpa/inet.h

extern uint32_t htonl(uint32_t);
extern uint16_t htons(uint16_t);
extern in_addr_t inet_addr(const char *__cp);
extern int inet_aton(const char *__cp, struct in_addr *__inp);
extern char *inet_ntoa(struct in_addr __in);
extern const char *inet_ntop(int __af, const void *__cp, char __buf, socklen_t __len);
extern int inet_pton(int __af, const char *__cp, void *__buf);
extern uint32_t ntohl(uint32_t);
extern uint16_t ntohs(uint16_t);
14.4.3 assert.h

#ifdef NDEBUG
#define assert(expr) ((__void)0)
#else
#define assert(expr)  ((__void) ((expr) ? 0 : (_assert_fail
(#expr, __FILE__, __LINE__, __PRETTY_FUNCTION__), 0)))
#endif

extern void __assert_fail(const char *__assertion, const char *
_file, unsigned int __line, const char
*__function);

14.4.4 cpio.h

#define C_IXOTH 000001
#define C_IWOTH 000002
#define C_IROTH 000004
#define C_IXGRP 000010
#define C_IWGRP 000020
#define C_IRGRP 000040
#define C_IXUSR 000100
#define C_IWUSR 000200
#define C_IRUSR 000400
#define C_ISVTX 001000
#define C_ISGID 002000
#define C_ISUID 004000
#define C_ISFIFO 010000
#define C_ISREG 0100000
#define C_ISCTG 0110000
#define C_ISLNK 0120000
#define C_ISSOCK 0140000
#define C_ISCHR 020000
#define C_ISDIR 040000
#define C_ISBLK 060000
#define MAGIC "070707"

14.4.5 ctype.h

extern const unsigned short **__ctype_b_loc(void);
extern const int32_t **__ctype_tolower_loc(void);
extern const int32_t **__ctype_toupper_loc(void);
extern int _tolower(int);
extern int _toupper(int);
extern int isalnum(int);
extern int isalnum_l(int c, locale_t locale);
extern int isalpha(int);
extern int isalpha_l(int c, locale_t locale);
extern int isascii(int __c);
extern int isblank(int);
extern int isblank_l(int c, locale_t locale);
extern int iscntrl(int);
extern int iscntrl_l(int c, locale_t locale);
extern int isdigit(int);
extern int isdigit_l(int c, locale_t locale);
extern int isgraph(int);
extern int isgraph_l(int c, locale_t locale);
extern int islower(int);
extern int islower_l(int c, locale_t locale);
extern int isprint(int);
extern int isprint_l(int c, locale_t locale);

© 2015 Linux Foundation
extern int ispunct(int);
extern int ispunct_l(int c, locale_t locale);
extern int isspace(int);
extern int isspace_l(int c, locale_t locale);
extern int isupper(int);
extern int isupper_l(int c, locale_t locale);
extern int isxdigit(int);
extern int isxdigit_l(int c, locale_t locale);
extern int toascii(int __c);
extern int tolower(int __c);
extern int tolower_l(int c, locale_t locale);
extern int toupper(int __c);
extern int toupper_l(int c, locale_t locale);

14.4.6 dirent.h

#define MAXNAMLEN NAME_MAX

typedef struct __dirstream DIR;

struct dirent {
    long int d_ino;
    off_t d_off;
    unsigned short d_reclen;
    unsigned char d_type;
    char d_name[256];
};

struct dirent64 {
    uint64_t d_ino;
    int64_t d_off;
    unsigned short d_reclen;
    unsigned char d_type;
    char d_name[256];
};

extern int alphasort(const struct dirent **__e1,
                      const struct dirent **__e2);

extern int alphasort64(const struct dirent64 **__e1,
                        const struct dirent64 **__e2);

extern int closedir(DIR * __dirp);
extern int dirfd(DIR * __dirp);
extern DIR *fopen64d(int __fd);
extern DIR *opendir(const char *__name);
extern struct dirent *readdir(DIR * __dirp);

extern struct dirent64 *readdir64(DIR * __dirp);
extern int readdir64_r(DIR * __dirp, struct dirent64 *__entry,
                        struct dirent64 **__result);

extern int readdir_r(DIR * __dirp, struct dirent *__entry,
                      struct dirent **__result);

extern void rewinddir(DIR * __dirp);
extern  int  scandir(const  char  *__dir,  struct  dirent  ***__namelist,
                     int (*__selector) (const struct dirent *),
                     int (*__cmp) (const struct dirent *,
                                   const struct dirent *));

extern int scandir64(const char *__dir, struct dirent64 ***__namelist,
                     int (*__selector) (const struct dirent64 *),
                     int (*__cmp) (const struct dirent64 *,
                                   const struct dirent64 *));

extern void seekdir(DIR * __dirp, long int __pos);
extern long int telldir(DIR * __dirp);
#define ELFMAG1 'E'
#define ELFMAG3 'F'
#define ELFMAG2 'L'
#define ELF64_R_INFO(sym, type) ((((Elf64_Word) (sym)) << 32) + (type))
#define ELF32_ST_INFO(bind, type) (((bind) << 4) + ((type) & 0xf))
#define ELF32_R_INFO(sym, type) (((sym) << 8) + ((type) & 0xff))
#define ELF32_M_INFO(sym, size) (((sym) << 8) + (unsigned char) (size))
#define ELF32_ST_BIND(val)      (((unsigned char) (val)) >> 4)
#define ELF64_R_TYPE(i) ((i) & 0xffffffff)
#define ELF64_R_SYM(i)  ((i) >> 32)
#define ELF32_M_SYM(info)       ((info) >> 8)
#define ELF32_ST_VISIBILITY(o)  ((o) & 0x03)
#define ELF32_M_SIZE(info)      ((unsigned char) (info))
#define ELF32_ST_TYPE(val)      ((val) & 0xf)
#define ELF32_R_TYPE(val)       ((val) & 0xff)
#define ELF32_R_SYM(val)        ((val) >> 8)
#define PF_X    (1 << 0)
#define SHF_WRITE       (1 << 0)
#define PF_W    (1 << 1)
#define SHF_ALLOC       (1 << 1)
#define PF_R    (1 << 2)
#define SHF_EXECINSTR   (1 << 2)
#define SHF_MERGE       (1 << 4)
#define SHF_STRINGS     (1 << 5)
#define SHF_INFO_LINK   (1 << 6)
#define SHF_LINK_ORDER  (1 << 7)
#define SHF_GROUP       (1 << 9)
#define EI_NIDENT       (16)
#define DT_ADDRTAGIDX(tag)      (DT_ADDRRNGHI - (tag))
#define DT_IA_64_PLT_RESERVE    (DT_LOPROC + 0)
#define DT_PPC64_GLINK (DT_LOPROC + 0)
#define DT_PPC64_GOT      (DT_LOPROC + 0)
#define DT_PPC64_OPD       (DT_LOPROC + 1)
#define DT_PPC64_OPD_SZ   (DT_LOPROC + 2)
#define DT_VALTAGIDX(tag)       (DT_VALRNGHI - (tag))
#define DT_VERSIONTAGIDX(tag)   (DT_VERNEEDNUM - (tag))
#define PT_IA_64_ARCHEXT (PT_LOPROC + 0)
#define PT_IA_64_UNWIND (PT_LOPROC + 1)
#define SHT_IA_64_EXT  (SHT_LOPROC + 0)
#define SHT_IA_64_UNWIND (SHT_LOPROC + 1)
#define DT_NULL 0
#define EI_MAG0 0
#define ELFCLASSNONE 0
#define ELFCLASSNONE 0
#define ELFOSABI_NONE 0
#define ELFOSABI_SYSV 0
#define ELF_NOTE_OS_LINUX 0
#define EM_NONE 0
#define ET_NONE 0
#define EV_NONE 0
#define PT_NULL 0
#define R_386_NONE 0
#define R_390_NONE 0
#define R_PPC_NONE 0
#define R_X86_64_NONE 0
#define SHN_UNDEF 0
#define SHT_NULL 0
#define STB_LOCAL 0

© 2015 Linux Foundation
#define STN_UNDEF       0
#define STT_NOTYPE      0
#define STV_DEFAULT     0
#define SYMINFO_NONE    0
#define R_IA64_NONE     0x00
#define DF_1_NOW        0x00000000
#define DF_ORIGIN       0x00000000
#define DF_P1_LAZYLOAD  0x00000000
#define DF_P1_PARINIT   0x00000000
#define EF_S390_HIGH_GPRS 0x00000000
#define DF_1_GLOBAL     0x00000000
#define DF_P1_GROUPPERM 0x00000000
#define DF_SYMBOLIC     0x00000000
#define DF_1_INITFIRST  0x00000000
#define DF_1_NOOPEN     0x00000000
#define DF_1_ORIGIN     0x00000000
#define DF_1_DIRECT     0x00000000
#define DF_1_TRANS      0x00000000
#define DF_1_INTERPOSE  0x00000000
#define DF_1_NODEFLIB   0x00000000
#define DF_1_DISPRELDNE 0x00000000
#define DF_1_DISPRELPND 0x00000000
#define SYMINFO_FLG_DIRECT      0x0001
#define SYMINFO_FLG_PASSTHRU    0x0002
#define SYMINFO_FLG_COPY        0x0004
#define SYMINFO_FLG_LAZYLOAD    0x0008
#define EF_CPU32        0x00810000
#define PF_MASKOS       0x0ff00000
#define SHF_MASKOS      0x0ff00000
#define GRP_COMDAT      0x1
#define SHF_IA_64_SHORT 0x10000000
#define SHF_IA_64_NORECOV 0x20000000
#define R_IA64_IMM14   0x21
#define R_IA64_IMM22   0x22
#define R_IA64_IMM64   0x23
#define R_IA64_DIR32MSB 0x24
#define R_IA64_DIR32LSB 0x25
#define R_IA64_DIR64MSB 0x26
#define R_IA64_DIR64LSB 0x27
#define R_IA64_GPREL22  0x2a
#define R_IA64_GPREL64I 0x2b
#define R_IA64_GPREL32MSB       0x2c
#define R_IA64_GPREL32LSB       0x2d
#define R_IA64_GPREL64MSB       0x2e
#define R_IA64_GPREL64LSB       0x2f
#define R_IA64_LTOFF22 0x32
#define R_IA64_LTOFF64I 0x33
#define R_IA64_PLTOFF22 0x3a
#define R_IA64_PLTOFF64I 0x3b
#define R_IA64_GPREL22 0x2a
#define R_IA64_GPREL64I 0x2f
#define R_IA64_LTOFF22 0x32
#define R_IA64_LTOFF64I 0x33
#define R_IA64_PLTOFF22 0x3a
#define R_IA64_PLTOFF64I 0x3b
#define R_IA64_PCREL60B 0x48
#define R_IA64_PCREL21B 0x49
#define R_IA64_PCREL21M 0x4a
#define R_IA64_PCREL21F 0x4b
#define R_IA64_PCREL32MSB 0x4c
#define R_IA64_PCREL32LSB 0x4d
#define R_IA64_PCREL64MSB 0x4e
#define R_IA64_PCREL64LSB 0x4f
#define R_IA64_LTOFF_FPTR22 0x52
#define R_IA64_LTOFF_FPTR64I 0x53
#define R_IA64_LTOFF_FPTR32MSB 0x54
#define R_IA64_LTOFF_FPTR32LSB 0x55
#define R_IA64_LTOFF_FPTR64MSB 0x56
#define R_IA64_LTOFF_FPTR64LSB 0x57
#define R_IA64_SEGREL32MSB 0x64
#define R_IA64_SEGREL32LSB 0x65
#define R_IA64_SEGREL64MSB 0x66
#define R_IA64_SEGREL64LSB 0x67
#define R_IA64_REL32MSB 0x6c
#define R_IA64_REL32LSB 0x6d
#define R_IA64_REL64MSB 0x6e
#define R_IA64_REL64LSB 0x6f
#define DT_HIOS 0x6ffff000
#define DT_VALRNGLO 0x6ffffd00
#define DT_GNU_PRELINKED 0x6ffffdf5
#define DT_GNU_CONFLICTSZ 0x6ffffdf6
#define DT_GNU_LIBLISTSZ 0x6ffffdf7
#define DT_CHECKSUM 0x6ffffdf8
#define DT_PLT_PADSZ 0x6ffffdf9
#define DT_MOVEENT 0x6ffffdfa
#define DT_MOVESZ 0x6ffffdff
#define DT_FEATURE_1 0x6ffffdfc
#define DT_POSFLAG_1 0x6ffffdff
#define DT_SYMINSZ 0x6ffffdfe
#define DT_SYMINT 0x6ffffdff
#define DT_VALRNGHI 0x6ffffdfe
#define DT_ADDRRNGLO 0x6fffff00
#define DT_GNU_HASH 0x6fffff05
#define DT_TLSDESC_PLT 0x6fffff06
#define DT_TLSDESC_GOT 0x6fffff07
#define DT_GNU_CONFLICT 0x6fffff08
#define DT_GNU_LIBLIST 0x6fffff09
#define DT_CONFIG 0x6fffff0a
#define DT_DEBUG 0x6fffff0b
#define DT_pes 0x6fffff0c
#define DT_GNU_STACK 0x6fffff0d
#define DT_GNU_RELRO 0x6fffff0e
#define DT_GNU_REL 0x6fffff0f
#define DT_PLT 0x6fffff10
#define DT_PIE 0x6fffff11
#define DT_TOC 0x6fffff12
#define DT_GNU_HASH 0x6fffff13
#define DT_TLSDESC_PIE 0x6fffff14
#define DT_SYMBOLIC 0x6fffff15
#define DT_GNU_PRELINKED 0x6fffff16
#define DT_GNU_REL 0x6fffff17
#define DT_GNU_RELRO 0x6fffff18
#define DT_SONAME 0x6fffff19
#define DT_RPATH 0x6fffff1a
#define DT_DYLD_INFO 0x6fffff1b
#define DT_RUNPATH 0x6fffff1c
#define DT_INIT 0x6fffff1d
#define DT_FINI 0x6fffff1e
#define DT_PLT_PIE 0x6fffff1f
#define DT_INIT_ARRAY 0x6fffff20
#define DT_FINI_ARRAY 0x6fffff21
#define DT_INIT_ARRAYestone 0x6fffff22
#define DT_FINI_ARRAYSTONE 0x6fffff23
#define DT_INTERP 0x6fffff24
#define DT_INIT 0x6fffff25
#define DT_FINI 0x6fffff26
#define DT_ENUMERATE 0x6fffff27
#define DT_REG 0x6fffff28
#define DT_PMOD 0x6fffff29
#define DT_GNU_STACK 0x6fffff2a
#define DT_GNU_REL 0x6fffff2b
#define DT_GNU_RELRO 0x6fffff2c
#define DT_SONAME 0x6fffff2d
#define DT_SYMBOLIC 0x6fffff2e
#define DT_GNU_BUILDID 0x6fffff2f
#define DTundefined 0x6fffff30
#define DT_REL 0x6fffff31
#define DT_PLT 0x6fffff32
#define DT_PIE 0x6fffff33
#define DT_SYMBOLIC 0x6fffff34
#define DT_GNU_PRELINKED 0x6fffff35
#define DT_GNU_REL 0x6fffff36
#define DT_GNU_RELRO 0x6fffff37
#define DT_SONAME 0x6fffff38
#define DT_SYMBOLIC 0x6fffff39
#define DT_GNU_BUILDID 0x6fffff3a
#define DTundefined 0x6fffff3b
#define DT-pic 0x6fffff3c
#define DT_GNU_STACK 0x6fffff3d
#define DT_GNU_REL 0x6fffff3e
#define DT_GNU_RELRO 0x6fffff3f
#define DT_SONAME 0x6fffff40
#define DT_SYMBOLIC 0x6fffff41
#define DT_GNU_BUILDID 0x6fffff42
#define DTundefined 0x6fffff43
#define DT-reloc 0x6fffff44
#define DT_GNU_STACK 0x6fffff45
#define DT_GNU_REL 0x6fffff46
#define DT_GNU_RELRO 0x6fffff47
#define DT_SONAME 0x6fffff48
#define DT_SYMBOLIC 0x6fffff49
#define DT_GNU_BUILDID 0x6fffff4a
#define DTundefined 0x6fffff4b
#define DT-gnudata 0x6fffff4c
#define DT_GNU_STACK 0x6fffff4d
#define DT_GNU_REL 0x6fffff4e
#define DT_GNU_RELRO 0x6fffff4f
#define DT_SONAME 0x6fffff50
#define DT_SYMBOLIC 0x6fffff51
#define DT_GNU_BUILDID 0x6fffff52
#define DTundefined 0x6fffff53
#define DT-merged 0x6fffff54
#define DT_GNU_STACK 0x6fffff55
#define DT_GNU_REL 0x6fffff56
#define DT_GNU_RELRO 0x6fffff57
#define DT_SONAME 0x6fffff58
#define DT_SYMBOLIC 0x6fffff59
#define DT_GNU_BUILDID 0x6fffff5a
#define DTundefined 0x6fffff5b
#define DT-absolute 0x6fffff5c
#define DT_GNU_STACK 0x6fffff5d
#define DT_GNU_REL 0x6fffff5e
#define DT_GNU_RELRO 0x6fffff5f
#define DT_SONAME 0x6fffff60
#define DT_SYMBOLIC 0x6fffff61
#define DT_GNU_BUILDID 0x6fffff62
#define DTundefined 0x6fffff63
#define DT-executable 0x6fffff64
#define DT_GNU_STACK 0x6fffff65
#define DT_GNU_REL 0x6fffff66
#define DT_GNU_RELRO 0x6fffff67
#define DT_SONAME 0x6fffff68
#define DT_SYMBOLIC 0x6fffff69
#define DT_GNU_BUILDID 0x6fffff6a
#define DTundefined 0x6fffff6b
#define DT-code 0x6fffff6c
#define DT_GNU_STACK 0x6fffff6d
#define DT_GNU_REL 0x6fffff6e
#define DT_GNU_RELRO 0x6fffff6f
#define DT_SONAME 0x6fffff70
#define DT_SYMBOLIC 0x6fffff71
#define DT_GNU_BUILDID 0x6fffff72
#define DTundefined 0x6fffff73
#define DT-unresolved 0x6fffff74
#define DT_GNU_STACK 0x6fffff75
#define DT_GNU_REL 0x6fffff76
#define DT_GNU_RELRO 0x6fffff77
#define DT_SONAME 0x6fffff78
#define DT_SYMBOLIC 0x6fffff79
#define DT_GNU_BUILDID 0x6fffff7a
#define DTundefined 0x6fffff7b
#define DT-external 0x6fffff7c
#define DT_GNU_STACK 0x6fffff7d
#define DT_GNU_REL 0x6fffff7e
#define DT_GNU_RELRO 0x6fffff7f
#define DT_SONAME 0x6fffff80
#define DT_SYMBOLIC 0x6fffff81
#define DT_GNU_BUILDID 0x6fffff82
#define DTundefined 0x6fffff83
#define DT-budget 0x6fffff84
#define DT_GNU_STACK 0x6fffff85
#define DT_GNU_REL 0x6fffff86
#define DT_GNU_RELRO 0x6fffff87
#define DT_SONAME 0x6fffff88
#define DT_SYMBOLIC 0x6fffff89
#define DT_GNU_BUILDID 0x6fffff8a
#define DTundefined 0x6fffff8b
#define DT-unknown 0x6fffff8c
#define DT_GNU_STACK 0x6fffff8d
#define DT_GNU_REL 0x6fffff8e
#define DT_GNU_RELRO 0x6fffff8f
#define DT_SONAME 0x6fffff90
#define DT_SYMBOLIC 0x6fffff91
#define DT_GNU_BUILDID 0x6fffff92
#define DTundefined 0x6fffff93
#define DT-struckthrough 0x6fffff94
#define DT_GNU_STACK 0x6fffff95
#define DT_GNU_REL 0x6fffff96
#define DT_GNU_RELRO 0x6fffff97
#define DT_SONAME 0x6fffff98
#define DT_SYMBOLIC 0x6fffff99
#define DT_GNU_BUILDID 0x6fffff9a
#define DTundefined 0x6fffff9b
#define DT-clear 0x6fffff9c
#define DT_GNU_STACK 0x6fffff9d
#define DT_GNU_REL 0x6fffff9e
#define DT_GNU_RELRO 0x6fffff9f
#define DT_SONAME 0x6fffffa0
#define DT_SYMBOLIC 0x6fffffa1
#define DT_GNU_BUILDID 0x6fffffa2
#define DTundefined 0x6fffffa3
#define DT-semantic 0x6fffffa4
#define DT_GNU_STACK 0x6fffffa5
#define DT_GNU_REL 0x6fffffa6
#define DT_GNU_RELRO 0x6fffffa7
#define DT_SONAME 0x6fffffa8
#define DT_SYMBOLIC 0x6fffffa9
#define DT_GNU_BUILDID 0x6fffffaa
#define DTundefined 0x6fffaaf
```c
#define DT_VERSYM       0x6ffffff0
#define SHT_GNU_ATTRIBUTES      0x6ffffff5
#define SHT_GNU_HASH     0x6ffffff6
#define SHT_GNU_LIBLIST  0x6ffffff7
#define SHT_CHECKSUM    0x6ffffff8
#define DT_RELACOUNT    0x6ffffff9
#define DT_RELCOUNT     0x6ffffffa
#define DT_FLAGS_1      0x6ffffffb
#define DT_VERDEF       0x6ffffffc
#define DT_VERDEFNUM    0x6ffffffd
#define SHT_GNU_LIBLIST  0x6ffffffe
#define DT_VERNEED      0x6fffffff
#define SHT_GNU_verdef  0x6fffffff
#define DT_VERNEEDNUM   0x6fffffff
#define PT_HIOS 0x6fffffff
#define SHT_GNU_versym  0x6fffffff
#define SHT_HIOS       0x6fffffff
#define DT_LOPROC       0x70000000
#define PT_LOPROC       0x70000000
#define SHT_LOPROC      0x70000000
#define R_IA64_LTV32MSB 0x74
#define R_IA64_LTV32LSB 0x75
#define R_IA64_LTV64MSB 0x76
#define R_IA64_LTV64LSB 0x77
#define R_IA64_PCREL21BI        0x79
#define R_IA64_PCREL22  0x7a
#define R_IA64_PCREL64I 0x7b
#define ELF_MAG0 0x7f
#define DT_AUXILIARY     0x7fffffff
#define DT_FILTER        0x7fffffff
#define DT_HIPROC       0x7fffffff
#define PT_HIPROC       0x7fffffff
#define SHT_HIPROC      0x7fffffff
#define R_IA64_IPLTMSB   0x80
#define R_IA64_IPLTLSB   0x81
#define R_IA64_COPY      0x84
#define R_IA64_SUB       0x85
#define R_IA64_LTOFF22X  0x86
#define R_IA64_LDXMOV    0x87
#define SHT_HIUSER       0x8fffffff
#define R_IA64_IPLFL22   0x91
#define R_IA64_DTPREL14  0xb1
#define R_IA64_DTPREL14I 0xb3
#define R_IA64_DTPREL22  0xb2
#define R_IA64_DTPREL22I 0xb3
#define R_IA64_DTPREL64MSB 0x96
#define R_IA64_DTPREL64LSB 0x97
#define R_IA64_LTOFF_TPREL22  0x9a
#define R_IA64_DTPMOD64MSB 0xa6
#define R_IA64_DTPMOD64LSB 0xa7
#define R_IA64_LTOFF_DTPMOD22 0xaa
#define R_IA64_DTPREL14I 0xb3
#define R_IA64_DTPREL22I 0xb3
#define R_IA64_DTPREL64I  0xb3
#define R_IA64_DTPREL32MSB 0xb4
#define R_IA64_DTPREL32LSB 0xb5
#define R_IA64_DTPREL64MSB 0xb6
#define R_IA64_DTPREL64LSB 0xb7
#define R_IA64_LTOFF_DTPREL22 0xba
#define PF_MASKPROC      0xf0000000
#define SHF_MASKPROC     0xf0000000
#define ET_LOOS 0xfe00
#define ET_HIOS 0xff00
#define ET_LOPROC      0xff00
#define SHN_LOPROC     0xff00
#define SHN_LORESERVE 0xff00
```
#define SYMINFO_BT_LOWRESERVE 0xff00
#define EF_IA_64_ARCH 0xff000000
#define SHN_HIPROC 0xfffff
#define SHN_LOOS 0xff20
#define SHN_HIOS 0xff3f
#define SHN_ABS 0xfff1
#define SHN_COMMON 0xfff2
#define SYMINFO_BT_PARENT 0xfffffe
#define ET_HIPROC 0xffff
#define PN_XNUM 0xfffff
#define SHN_HIRESERVE 0xfffff
#define SHN_XINDEX 0xfffff
#define SYMINFO_BT_SELF 0xfffff
#define DT_IA_64_NUM 1
#define DT_NEEDED 1
#define DT_PPC_NUM 1
#define EI_MAG1 1
#define ELFCLASS32 1
#define ELFDATA2LSB 1
#define ELF_NOTE_OS_GNU 1
#define ET_REL 1
#define EV_CURRENT 1
#define NT_GNU_ABI_TAG 1
#define DT_LOAD 1
#define R_386_32 1
#define R_390_8 1
#define R_PPC_ADDR32 1
#define R_PPC_ADDR32 1
#define R_X86_64_64 1
#define SHT_PROGBITS 1
#define STB_GLOBAL 1
#define STT_OBJECT 1
#define STV_INTERNAL 1
#define SYMINFO_CURRENT 1
#define DT_STRSZ 10
#define R_386_GOTPC 10
#define R_390_GLOB_DAT 10
#define R_PPC_REL24 10
#define R_X86_64_32 10
#define SHT_SHLIB 10
#define STB_LOOS 10
#define STT_GNU_IFUNC 10
#define STT_LOOS 10
#define R_PPC64_TPREL16_HIGHESTA 100
#define R_PPC64_TPREL16_HIGHESTA 100
#define R_PPC64_TPREL16_HIGHESTA 100
#define R_PPC64_TPREL16_HIGHESTA 100
#define DT_ADDRNUM 11
#define DT_SYMNT 11
#define R_386_32PLT 11
#define R_390_JMP_SLOT 11
#define R_PPC_REL14 11
#define R_X86_64_32S 11
#define SHT_DYNSYM 11
#define DT_INIT 12
#define DT_VALNUM 12
#define R_390_RELATIVE 12
#define R_PPC_REL14_BRTAKEN 12
#define R_X86_64_16 12
#define STB_HIOS 12
#define STT_HIOS 12
#define DT_FINI 13
#define R_390_GOTOFF32 13
#define R_PPC_REL14_BRNTAKEN 13
#define R_PPC_REL14_BRNTAKEN 13
#define R_X86_64_PC16 13
#define STB_LOPROC 13
#define STT_LOPROC 13
#define DT_SONAME 14
#define R_386_TLS_TPOFF 14
#define R_390_GOTPC 14
#define R_PPC_GOT16 14
#define R_X86_64_8 14
#define SHT_INIT_ARRAY 14
#define DT_RPATH 15
#define R_386_TLS_IE 15
#define R_390_GOT16 15
#define R_PPC_GOT16_L0 15
#define R_X86_64_PC8 15
#define SHT_FINI_ARRAY 15
#define STB_HIPROC 15
#define STT_HIPROC 15
#define DT_SYMBOLIC 16
#define DT_VERSIONTAGNUM 16
#define R_386_TLS_GOTIE 16
#define R_390_PC16 16
#define R_PPC_GOT16_HI 16
#define R_X86_64_DTPMOD64 16
#define SHT_PREINIT_ARRAY 16
#define DT_REL 17
#define R_386_TLS_LE 17
#define R_390_PC16DBL 17
#define R_PPC_GOT16_FA 17
#define R_X86_64_DTPOFF64 17
#define SHT_GROUP 17
#define DT_RELSZ 18
#define R_386_TLS_GD 18
#define R_390_PC32DBL 18
#define R_PPC_COPY 18
#define R_X86_64_TPOFF64 18
#define SHT_SYMTAB_SHNDX 18
#define DT_RELENT 19
#define R_386_16 19
#define R_390_12 19
#define R_PPC_ADDR24 19
#define R_X86_64_PC32 19
#define SHT_SYMTAB 19
#define STB_WEAK 19
#define STT_FUNC 19
#define STV_HIDDEN 19
#define SYMINFO_NUM 19
#define DT_PLTREL 20
#define EM_PPC 20
#define R_386_16 20
#define R_390_PC32DBL 20
#define R_PPC_ADDR24 20
#define R_X86_64_PC32 20
#define SHT_SYMTAB 20
#define STB_WEAK 20
#define STT_FUNC 20
#define STV_HIDDEN 20
#define SYMINFO_NUM 20
#define DT_PLTREL 20
#define EM_PPC 20
#define R_386_16 20
#define R_390_PLT32DBL 20
#define R_PPC_GLOB_DAT 20
#define R_X86_64_TLSLD 20
#define DT_DEBUG 21
#define EM_PPC64 21
#define R_386_PC16 21
#define R_390_GOTPCDBL 21
#define R_PPC_JMP_SLOT 21
#define R_X86_64_DTPOFF32 21
#define DT_TEXTREL 22
#define EM_S390 22
#define R_386_8 22
#define R_390_64 22
#define R_PPC_RELATIVE 22
#define R_X86_64_GOTTOFF 22
#define DT_JMPREL 23
#define R_386_PC8 23
#define R_390_PC64 23
#define R_PPC_LOCAL24PC 23
#define R_X86_64_TPOFF32 23
#define DT_BIND_NOW 24
#define R_386_TLS_GD_32 24
#define R_390_GOT64 24
#define R_PPC_UADDR32 24
#define R_X86_64_PC64 24
#define R_PPC64_JMP_IREL 247
#define R_PPC64_IRELATIVE 248
#define R_PPC64_REL16 249
#define R_PPC_REL16 249
#define DT_INIT_ARRAY 25
#define R_386_TLS_GD_PUSH 25
#define R_390_PLT64 25
#define R_PPC_UADDR16 25
#define R_X86_64_GOTOFF64 25
#define R_PPC64_REL16_LO 250
#define R_PPC64_REL16_HI 251
#define R_PPC_REL16_HI 251
#define R_PPC64_REL16_HA 252
#define R_PPC_REL16_HA 252
#define R_PPC_TOC16 255
#define DT_FINI_ARRAY 26
#define R_386_TLS_GD_CALL 26
#define R_390_GOTENT 26
#define R_PPC_REL32 26
#define R_X86_64_GOTPC32 26
#define DT_INIT_ARRAYSZ 27
#define R_386_TLS_GD_POP 27
#define R_390_GOTOFF16 27
#define R_PPC_PLT32 27
#define R_X86_64_GOT64 27
#define DT_FINI_ARRAYSZ 28
#define R_386_TLS_LDM_32 28
#define R_390_GOTOFF64 28
#define R_PPC_PTREL32 28
#define R_X86_64_GOTPCREL64 28
#define DT_RUNPATH 29
#define R_386_TLS_LDM_PUSH 29
#define R_390_GOTPLT12 29
#define R_PPC_PLT16_LO 29
#define R_X86_64_GOTPC64 29
#define DT_EXTRANUM 3
#define DT_PLTGOT 3
#define DT_PPC64_NUM 3
#define EI_MAG3 3
#define ELFCLASSNUM 3
#define ELFOSABI_LINUX 3
#define ELF_NOTE_OS_FREEBSD 3
#define EM_386 3
```c
#define ET_DYN 3
#define PT_INTERP 3
#define R_386_GOT32 3
#define R_390_16 3
#define R_PPC_ADDR16 3
#define R_X86_64_GOT32 3
#define SHT_STRTAB 3
#define STB_NUM 3
#define STT_SECTION 3
#define STV_PROTECTED 3
#define DT_FLAGS 30
#define R_386_TLS_LDM_CALL 30
#define R_390_GOTPLT16 30
#define R_PPC_PPT16_HI 30
#define R_X86_64_GOTPLT64 30
#define R_386_TLS_LDM_POP 31
#define R_390_GOTPLT32 31
#define R_PPC_PPT16_HA 31
#define R_X86_64_PPTOFF64 31
#define DT_ENCODING 32
#define R_386_TLS_LDM_32 32
#define R_390_GOTPLT64 32
#define R_PPC_SPEAR16 32
#define R_X86_64_SIZE32 32
#define DT_PREINIT_ARRAYSZ 33
#define R_386_TLS_IE 33
#define R_390_GOTPLTENT 33
#define R_PPC_SPEAR16_HI 33
#define R_X86_64_SIZE64 33
#define DT_NUM 34
#define R_386_TLS_LE 34
#define R_390_PPTOFF32 34
#define R_PPC_SPEAR16_L0 34
#define R_X86_64_GOTPC32_TLSDESC 34
#define R_386_TLS_DTPMOD32 35
#define R_390_PPTOFF32 35
#define R_PPC_SPEAR16_HI 35
#define R_X86_64_TLSDESC_CALL 35
#define R_386_TLS_DTPOFF32 36
#define R_390_PPTOFF64 36
#define R_PPC_SPEAR16_HA 36
#define R_X86_64_TLSDESC 36
#define R_386_TLS_TPOFF32 37
#define R_390_PPTOFF64 37
#define R_PPC_SPEAR16_HA 37
#define R_X86_64_TLSDESC 37
#define R_386_TLS_LLOAD 37
#define R_PPC64_ADDR30 37
#define R_X86_64_IRELATIVE 37
#define R_390_TLS_GDCALL 38
#define R_PPC64_ADDR64 38
#define R_X86_64_NUM 38
#define R_386_TLS_GOTDESC 39
#define R_390_TLS_LDCALL 39
#define R_PPC64_ADDR16_HIGHER 39
#define DT_HASH 4
#define EI_CLASS 4
#define ET_CORE 4
#define PT_NOTE 4
#define R_386_PLT32 4
#define R_390_32 4
#define R_PPC_ADDR16_L0 4
#define R_X86_64_PLT32 4
#define SELFMAG 4
#define SHT_RELA 4
#define STT_FILE 4
#define EM_ARM 40
#define R_386_TLS_DESC_CALL 40
```
#define R_390_TLS_GD32  40
#define R_PPC64_ADDR16_HIGHERA  40
#define R_386_TLS_DESC  41
#define R_390_TLS_GD64  41
#define R_PPC64_ADDR16_HIGHEST  41
#define R_386_IRELATIVE 42
#define R_390_TLS_GOTIE12       42
#define R_PPC64_ADDR16_HIGESTA 42
#define R_386_NUM 43
#define R_390_TLS_GOTIE32       43
#define R_PPC64_UADDR64 43
#define R_390_TLS_GOTIE64       44
#define R_PPC64_PLT64 45
#define R_390_TLS_GDM32 45
#define R_PPC64_PPT64 46
#define R_390_TLS_IE32  47
#define R_PPC64_PPTIE64 47
#define R_390_TLS_IE64  48
#define R_PPC64_TOC16 48
#define R_390_TLS_IEENT 49
#define R_PPC64_TOC16_HI 49
#define DT_STRTAB       5
#define EI_DATA 5
#define ET_NUM 5
#define PT_SHLIB 5
#define R_386_COPY 5
#define R_390_PC32 5
#define R_PPC_ADDR16_HI 5
#define R_X86_64_COPY 5
#define SHT_HASH 5
#define STT_COMMON 5
#define EM_IA_64 50
#define R_390_TLS_LE32  50
#define R_PPC64_TOC16_HA 50
#define R_390_TLS_LE64  51
#define R_PPC64_TOC16 51
#define R_390_TLS_LE64  52
#define R_PPC64_LD032 52
#define R_PPC64_PPLT16 52
#define R_390_TLS_LD064 53
#define R_PPC64_PPTGOT16_L0 53
#define R_390_TLS_DTPMOD 54
#define R_PPC64_PPTGOT16_HI 54
#define R_390_TLS_DTPPOFF 55
#define R_PPC64_PPTGOT16_HA 55
#define R_390_TLS_TPOFF 56
#define R_PPC64_PPTGOT16_HA 56
#define R_390_TLS_TPOFF 57
#define R_PPC64_ADDR16_DS 57
#define R_390_20 57
#define R_PPC64_ADDR16_L0_DS 58
#define R_390_GOT20 58
#define R_PPC64_GOTPLT20 59
#define R_390_GOT between 59
#define R_390_TLS_TPOFF 60
#define R_PPC64_PPT16_L0_DS 61
#define R_390_NUM 61
#define R_PPC64_SECTOFF_DS  61
#define EM_X86_64  62
#define R_PPC64_SECTOFF_L0_DS  62
#define R_PPC64_TOC16_DS  63
#define R_PPC64_TOC16_L0_DS  64
#define R_PPC64_PLTGOT16_DS  65
#define R_PPC64_PLTGOT16_L0_DS  66
#define R_PPC64_TLS  67
#define R_PPC_TLS  67
#define R_PPC64_DTPMOD64  68
#define R_PPC_DTPMOD32  68
#define R_PPC64_TPREL16  69
#define R_PPC_TPREL16  69
#define DT_RELA  7
#define EI_OSABI  7
#define PT_TLS  7
#define R_386_JMP_SLOT  7
#define R_390_GOT32  7
#define R_PPC_ADDR14  7
#define R_X86_64_JUMP_SLOT  7
#define SHT_NOTE  7
#define STT_NUM  7
#define R_PPC64_TPREL16_L0  70
#define R_PPC_TPREL16_L0  70
#define R_PPC64_TPREL16_HI  71
#define R_PPC_TPREL16_HI  71
#define R_PPC64_TPREL16_HA  72
#define R_PPC_TPREL16_HA  72
#define R_PPC64_TPREL64  73
#define R_PPC_TPREL32  73
#define R_PPC64_DTPREL16  74
#define R_PPC_DTPREL16  74
#define R_PPC64_DTPREL16_L0  75
#define R_PPC_DTPREL16_L0  75
#define R_PPC64_DTPREL16_HI  76
#define R_PPC_DTPREL16_HI  76
#define R_PPC64_DTPREL16_HA  77
#define R_PPC_DTPREL16_HA  77
#define R_PPC64_DTPREL64  78
#define R_PPC_DTPREL32  78
#define R_PPC64_GOT_TLSGD16  79
#define R_PPC_GOT_TLSGD16  79
#define DT_RELASZ  8
#define EI_ABIVERSION  8
#define PT_NUM  8
#define R_386_RELATIVE  8
#define R_390_PLT32  8
#define R_PPC_ADDR14_BRTAKEN  8
#define R_X86_64_RELATIVE  8
#define SHT_NOBITS  8
#define R_PPC64_GOT_TLSGD16_LO  80
#define R_PPC_GOT_TLSGD16_LO  80
#define R_PPC64_GOT_TLSGD16_HI  81
#define R_PPC_GOT_TLSGD16_HI  81
#define R_PPC64_GOT_TLSGD16_HA  82
#define R_PPC_GOT_TLSGD16_HA  82
#define R_PPC64_GOT_TLSLD16  83
#define R_PPC_GOT_TLSLD16  83
#define R_PPC64_GOT_TLSLD16_LO  84
#define R_PPC_GOT_TLSLD16_LO  84
#define R_PPC64_GOT_TLSLD16_HI  85
#define R_PPC_GOT_TLSLD16_HI  85
#define R_PPC64_GOT_TLSLD16_HA  86
#define R_PPC_GOT_TLSLD16_HA  86
#define R_PPC64_GOT_TPREL16_DS  87
#define R_PPC_GOT_TPREL16  87
#define R_PPC64_GOT_TPREL16_L0_DS    88
#define R_PPC64_GOT_TPREL16_L0      88
#define R_PPC64_GOT_TPREL16_HI     89
#define R_PPC64_GOT_TPREL16_HI     89
#define DT_RELAENT          9
#define EI_PAD             9
#define R_386_GOTOFF        9
#define R_390_COPY          9
#define R_PPC_ADDR14_BRNTAKEN 9
#define R_X86_64_GOTPCREL   9
#define SHT_REL            9
#define R_PPC64_GOT_TPREL16_HA  90
#define R_PPC64_GOT_TPREL16_HI  90
#define R_PPC64_GOT_DTPREL16_DS 91
#define R_PPC64_GOT_DTPREL16_HI 91
#define R_PPC64_GOT_DTPREL16_L0_DS 92
#define R_PPC64_GOT_DTPREL16_L0  92
#define R_PPC64_GOT_DTPREL16_HI  93
#define R_PPC64_GOT_DTPREL16_HI  93
#define R_PPC64_GOT_DTPREL16_HA  94
#define R_PPC64_TPREL16_DS     95
#define R_PPC64_TPREL16_LO_DS   96
#define R_PPC64_TPREL16_HIGHER 97
#define R_PPC64_TPREL16_HIGHERA 98
#define R_PPC64_TPREL16_HIGHEST 99
#define ELF64_M_INFO(sym, size) ELF32_M_INFO (sym, size)
#define ELF64_M_SIZE(info) ELF32_M_SIZE (info)
#define ELF64_M_SYM(info) ELF32_M_SYM (info)
#define ELF64_ST_BIND(val) ELF32_ST_BIND (val)
#define ELF64_ST_INFO(bind, type) ELF32_ST_INFO ((bind), (type))
#define ELF64_ST_TYPE(val) ELF32_ST_TYPE (val)
#define ELF64_ST_VISIBILITY(o) ELF32_ST_VISIBILITY(o)
#define ELF_NOTE_GNU "GNU"
#define ELF_NOTE_ABI NT_GNU_ABI_TAG
#define R_PPC64_ADDR14 BRNTAKEN R_PPC_ADDR14
#define R_PPC64_ADDR14_BRTaken R_PPC_ADDR14_BRTaken
#define R_PPC64_ADDR16          R_PPC_ADDR16
#define R_PPC64_ADDR16_HA       R_PPC_ADDR16_HA
#define R_PPC64_ADDR16_HI       R_PPC_ADDR16_HI
#define R_PPC64_ADDR16_LO       R_PPC_ADDR16_LO
#define R_PPC64_ADDR24          R_PPC_ADDR24
#define R_PPC64_ADDR32          R_PPC_ADDR32
#define R_PPC64_COPY            R_PPC_COPY
#define R_PPC64_GLOB_DAT        R_PPC_GLOB_DAT
#define R_PPC64_GOT16           R_PPC_GOT16
#define R_PPC64_GOT16_HI        R_PPC_GOT16_HI
#define R_PPC64_GOT16_L0        R_PPC_GOT16_L0
#define R_PPC64_JMP_SLOT        R_PPC_JMP_SLOT
#define R_PPC64_NONE            R_PPC_NONE
#define R_PPC64_PLT16_HA        R_PPC_PLT16_HA
#define R_PPC64_PLT16_HI        R_PPC_PLT16_HI
#define R_PPC64_PLT16_L0        R_PPC_PLT16_L0
#define R_PPC64_PLT32           R_PPC_PLT32
#define R_PPC64_PLTREL32        R_PPC_PLTREL32
#define R_PPC64_REL14           R_PPC_REL14
#define R_PPC64_REL14_BRNTAKEN  R_PPC_REL14_BRNTaken
#define R_PPC64_REL14_BRTaken   R_PPC_REL14_BRTaken
#define R_PPC64_REL24           R_PPC_REL24
#define R_PPC64_REL32           R_PPC_REL32
#define R_PPC64_RELATIVE        R_PPC_RELATIVE
#define R_PPC64_SECTOFF         R_PPC_SECTOFF
#define R_PPC64_SECTOFF_HA      R_PPC_SECTOFF_HA
```c
#define R_PPC64_SECTOFF_HI     R_PPC_SECTOFF_HI
#define R_PPC64_SECTOFF_LO     R_PPC_SECTOFF_LO
#define R_PPC64_UADDR16        R_PPC_UADDR16
#define R_PPC64_UADDR32        R_PPC_UADDR32
#define ELFMAG        "\177ELF"

typedef uint32_t Elf32_Addr;
typedef uint64_t Elf64_Addr;
typedef uint32_t Elf32_Word;
typedef uint64_t Elf64_Word;
typedef int32_t Elf32_Sword;
typedef int64_t Elf64_Sword;
typedef uint64_t Elf32_Xword;
typedef uint64_t Elf64_Xword;
typedef int64_t Elf32_Sxword;
typedef int64_t Elf64_Sxword;
typedef uint32_t Elf32_Off;
typedef uint64_t Elf64_Off;
typedef struct {
    Elf32_Word p_type;          /* Segment type */
    Elf32_Off p_offset;         /* Segment file offset */
    Elf32_Addr p_vaddr;         /* Segment virtual address */
    Elf32_Addr p_paddr;         /* Segment physical address */
    Elf32_Word p_filesz;        /* Segment size in file */
    Elf32_Word p_memsz;         /* Segment size in memory */
    Elf32_Word p_flags;         /* Segment flags */
    Elf32_Word p_align;         /* Segment alignment */
} Elf32_Phdr;
typedef struct {
    Elf64_Word p_type;          /* Segment type */
    Elf64_Word p_flags;         /* Segment flags */
    Elf64_Off p_offset;         /* Segment file offset */
    Elf64_Addr p_vaddr;         /* Segment virtual address */
    Elf64_Addr p_paddr;         /* Segment physical address */
    Elf64_Xword p_filesz;       /* Segment size in file */
    Elf64_Xword p_memsz;        /* Segment size in memory */
    Elf64_Xword p_align;        /* Segment alignment */
} Elf64_Phdr;
typedef uint16_t Elf32_Half;
typedef uint16_t Elf64_Half;
typedef uint16_t Elf32_Section;
typedef uint16_t Elf64_Section;
typedef struct {
    Elf32_Word n_namesz;
    Elf32_Word n_descsz;
    Elf32_Word n_type;
} Elf32_Nhdr;
typedef struct {
    Elf64_Word n_namesz;
    Elf64_Word n_descsz;
    Elf64_Word n_type;
} Elf64_Nhdr;
typedef struct {
    Elf64_Word st_name;
    unsigned char st_info;
    unsigned char st_other;
    Elf64_Section st_shndx;
    Elf64_Addr st_value;
    Elf64_Xword st_size;
} Elf64_Sym;
typedef struct {
    Elf32_Word st_name;
    Elf32_Addr st_value;
    Elf32_Word st_size;
    unsigned char st_info;
    unsigned char st_other;
```
typedef struct {
Elf32_Addr r_offset;
Elf64_Word r_info;
} Elf64_Rel;
typedef struct {
Elf32_Addr r_offset;
Elf64_Word r_info;
Elf64_Sxword r_addend;
} Elf64_Rela;
typedef struct {
Elf32_Half vd_version;
Elf32_Half vd_flags;
Elf32_Half vd_ndx;
Elf32_Word vd_cnt;
Elf64_Word vd_hash;
Elf64_Word vd_aux;
Elf64_Word vd_next;
} Elf64_Verdef;
typedef struct {
Elf64_Half vd_version;
Elf64_Half vd_flags;
Elf32_Half vd_ndx;
Elf32_Half vd_cnt;
Elf64_Word vd_hash;
Elf64_Word vd_aux;
Elf64_Word vd_next;
} Elf64_Verneed;
typedef struct {
Elf32_Word vna_hash;
Elf32_Half vna_flags;
Elf32_Half vna_other;
Elf64_Word vna_name;
Elf64_Word vna_next;
} Elf64_Verdaux;

LSB Core - Generic 5.0
typedef struct {
    Elf64_Word vna_hash;
    Elf64_Half vna_flags;
    Elf64_Half vna_other;
    Elf64_Word vna_name;
    Elf64_Word vna_next;
} Elf64_Vernaux;

typedef struct {
    unsigned char e_ident[EI_NIDENT];
    Elf64_Half e_type;
    Elf64_Half e_machine;
    Elf64_Word e_version;
    Elf64.Addr e_entry;
    Elf64_Off e_phoff;
    Elf64_Off e_shoff;
    Elf64_Word e_flags;
    Elf64_Half e_ehsize;
    Elf64_Half e_phentsize;
    Elf64_Half e_phnum;
    Elf64_Half e_shentsize;
    Elf64_Half e_shnum;
    Elf64_Half e_shstrndx;
} Elf64_Ehdr;

typedef struct {
    unsigned char e_ident[EI_NIDENT];
    Elf32_Half e_type;
    Elf32_Half e_machine;
    Elf32_Word e_version;
    Elf32.Addr e_entry;
    Elf32_Off e_phoff;
    Elf32_Off e_shoff;
    Elf32_Word e_flags;
    Elf32_Half e_ehsize;
    Elf32_Half e_phentsize;
    Elf32_Half e_phnum;
    Elf32_Half e_shentsize;
    Elf32_Half e_shnum;
    Elf32_Half e_shstrndx;
} Elf32_Ehdr;

typedef struct {
    Elf32_Word sh_name;
    Elf32_Word sh_type;
    Elf32_Word sh_flags;
    Elf32_Addr sh_addr;
    Elf32_Off sh_offset;
    Elf32_Word sh_size;
    Elf32_Word sh_link;
    Elf32_Word sh_info;
    Elf32_Word sh_addralign;
    Elf32_Word sh_entsize;
} Elf32_Shdr;

typedef struct {
    Elf64_Word sh_name;
    Elf64_Word sh_type;
    Elf64_Xword sh_flags;
    Elf64_Addr sh_addr;
    Elf64_Off sh_offset;
    Elf64_Xword sh_size;
    Elf64_Word sh_link;
    Elf64_Word sh_info;
    Elf64_Xword sh_addralign;
    Elf64_Xword sh_entsize;
} Elf64_Shdr;

typedef struct {
    Elf32_Sword d_tag;
} Elf32_Shdr;

typedef struct {
    Elf64_Word sh_name;
    Elf64_Word sh_type;
    Elf64_Xword sh_flags;
    Elf64_Addr sh_addr;
    Elf64_Off sh_offset;
    Elf64_Xword sh_size;
    Elf64_Word sh_link;
    Elf64_Word sh_info;
    Elf64_Xword sh_addralign;
    Elf64_Xword sh_entsize;
} Elf64_Shdr;

typedef struct {
    Elf32_Sword d_tag;
union {
    Elf32_Word d_val;
    Elf32_Addr d_ptr;
} d_un;
} Elf32_Dyn;
typedef struct {
    Elf64_Sxword d_tag;
    union {
        Elf64_Xword d_val;
        Elf64_Addr d_ptr;
    } d_un;
} Elf64_Dyn;

14.4.8 endian.h

#define __LITTLE_ENDIAN 1234
#define __BIG_ENDIAN 4321
#define BIG_ENDIAN __BIG_ENDIAN
#define BYTE_ORDER __BYTE_ORDER
#define LITTLE_ENDIAN __LITTLE_ENDIAN

14.4.9 envz.h

extern error_t envz_add(char **envz, size_t * envz_len, const char *name,
                        const char *value);
extern char envz_entry(const char *envz, size_t envz_len,
                       const char *name);
extern char envz_get(const char *envz, size_t envz_len, const char *name);
extern error_t envz_merge(char **envz, size_t * envz_len,
                           const char *envz2, size_t envz2_len,
                           int override);
extern void envz_remove(char **envz, size_t * envz_len, const char *name);
extern void envz_strip(char **envz, size_t * envz_len);

14.4.10 err.h

extern void err(int eval, const char *fmt, ...);
extern void errx(int eval, const char *fmt, ...);
extern void verrx(int eval, const char *fmt, va_list args);
extern void warn(const char *fmt, ...);
extern void warnx(const char *fmt, ...);

14.4.11 errno.h

#define errno (*__errno_location())
#define EPERM  1               /* Operation not permitted */
#define ECHILD 10              /* No child processes */
#define ENETDOWN  100           /* Network is down */
#define ENETUNREACH  101        /* Network is unreachable */
#define ENETRESET  102          /* Network dropped connection because of reset */
#define ECONNRESET 104          /* Connection reset by peer */
#define ENOBUFFS 105            /* No buffer space available */
#define EISCONN 106             /* Transport endpoint is already
#define ENOTCONN 107 /* Transport endpoint is not connected */
#define ESHUTDOWN 108 /* Cannot send after transport endpoint shutdown */
#defineETOOMANYREFS 109 /* Too many references: cannot splice */
#define EAGAIN 11 /* Try again */
#define ETIMEDOUT 110 /* Connection timed out */
#define ECONNRUFUSED 111 /* Connection refused */
#define EHOSTDOWNow / * Host is down */
#define EHOSTUNREACH 113 /* No route to host */
#define EALREADY 114 /* Operation already in progress */
#define EINPROGRESS 115 /* Operation now in progress */
#define ESTALE 116 /* Stale NFS file handle */
#define EUCLEAN 117 /* Structure needs cleaning */
#define ENOTNAM 118 /* Not a XENIX named type file */
#define ENAVAIL 119 /* No XENIX semaphores available */
#define ENOMEM 12 /* Out of memory */
#define EISNAM 120 /* Is a named type file */
#define ERMTIEI0 121 /* Remote I/O error */
#define EDQUOT 122 /* Quota exceeded */
#define ENOMEDIUM 123 /* No medium found */
#define EHOSTDOWN 124 /* Wrong medium type */
#define ECANCELED 125 /* Operation Canceled */
#define ENOTRECOVERABLE 126 /* State not recoverable */
#define EREMOTEIO 127 /* Operation not possible due to RF-kill */
#define EFAULT 128 /* Bad address */
#define ENOTBLK 129 /* Block device required */
#define EBUSY 130 /* Device or resource busy */
#define EXIST 131 /* File exists */
#define EXDEV 132 /* Cross-device link */
#define ENDEV 133 /* No such device */
#define ENOENT 134 /* No such file or directory */
#define ENOTDIR 135 /* Not a directory */
#define EISDIR 136 /* Is a directory */
#define EINVAL 137 /* Invalid argument */
#define EMFILE 138 /* Invalid argument */
#define ENFILE 139 /* File table overflow */
#define EMFILE 140 /* Too many open files */
#define ENOTTTY 141 /* Not a typewriter */
#define ETEXTBSY 142 /* Text file busy */
#define EFBIG 143 /* File too large */
#define EPIPE 144 /* Illegal seek */
#define ESRC 145 /* No such process */
#define EROFS 146 /* Read-only file system */
#define EMLINK 147 /* Too many links */
#define EPIPE 148 /* Broken pipe */
#define EDOM 149 /* Math argument out of domain of func */
#define ERANGE 150 /* Math result not representable */
#define EDEADLK 151 /* Resource deadlock would occur */
#define ENAMETOOLONG 152 /* File name too long */
#define ENOLCK 153 /* No record locks available */
#define ENOSYS 154 /* Function not implemented */
#define ENOTEMPTY 155 /* Directory not empty */
#define EINTR 156 /* Interrupted system call */
#define ELOOP 157 /* Too many symbolic links encountered */
#define ENOMSG 42 /* No message of desired type */
#define EIDRM 43 /* Identifier removed */
#define ECHRNG 44 /* Channel number out of range */
#define EL2NSYNC 45 /* Level 2 not synchronized */
#define EL3HLT 46 /* Level 3 halted */
#define EL3RST 47 /* Level 3 reset */
#define ELNRNG 48 /* Link number out of range */
#define EUNATCH 49 /* Protocol driver not attached */
#define EIO 5 /* I/O error */
#define ENOANO 55 /* No anode */
#define EBADRQc 56 /* Invalid request code */
#define EBADSLT 57 /* Invalid slot */
#define EBFONT 59 /* Bad font file format */
#define ENUMXIO 6 /* No such device or address */
#define ENODATA 61 /* No data available */
#define ETIME 62 /* Timer expired */
#define ENONET 64 /* Machine is not on the network */
#define ENOPKG 65 /* Package not installed */
#define ENOLINK 66 /* Object is remote */
#define ENOLINK 67 /* Link has been severed */
#define EADV 68 /* Advertise error */
#define ESRMNT 69 /* Srmount error */
#define E2BIG 7 /* Argument list too long */
#define ECOMM 70 /* Communication error on send */
#define EPROTOK 71 /* Protocol error */
#define EMULTIHOP 72 /* Multihop attempted */
#define EDOTDOT 73 /* RFS specific error */
#define ESRMNT 74 /* Not a data message */
#define EOVERFLOW 75 /* Value too large for defined data type */
#define EPROTONOSUPPORT 76 /* Name not unique on network */
#define EPROTONOSUPPORT 77 /* File descriptor in bad state */
#define ENOMEM 78 /* Remote address changed */
#define ENOSPTOSUPPORT 79 /* Can not access a needed shared library */
#define ENOEXEC 80 /* Exec format error */
#define EPROTONOSUPPORT 81 /* Accessing a corrupted shared library */
#define EPROTONOSUPPORT 82 /* .lib section in a.out corrupted */
#define EPROTONOSUPPORT 83 /* Attempting to link in too many shared libraries */
#define EPROTONOSUPPORT 84 /* Cannot exec a shared library directly */
#define EPROTONOSUPPORT 85 /* Illegal byte sequence */
#define EPROTONOSUPPORT 86 /* Interrupted system call should be restarted */
#define EPROTONOSUPPORT 87 /* Streams pipe error */
#define EPROTONOSUPPORT 88 /* Too many users */
#define EPROTONOSUPPORT 89 /* Socket operation on non-socket */
#define EPROTONOSUPPORT 90 /* Destination address required */
#define EPROTONOSUPPORT 91 /* Bad file number */
#define EPROTONOSUPPORT 92 /* Message too long */
#define EPROTONOSUPPORT 93 /* Protocol wrong type for socket */
#define EPROTONOSUPPORT 94 /* Protocol not available */
#define EPROTONOSUPPORT 95 /* Socket type not supported */
#define EPROTONOSUPPORT 96 /* Operation not supported on
transport endpoint */
#define EPFNOSUPPORT 96 /* Protocol family not supported */
#define EAFNOSUPPORT 97 /* Address family not supported by protocol */
#define EADDRINUSE 98 /* Address already in use */
#define EADDRNOTAVAIL 99 /* Cannot assign requested address */
#define EWOULDBLOCK EAGAIN /* Operation would block */
#define ENOTSUP EOPNOTSUPP

extern int *__errno_location(void);

14.4.12 error.h

extern void error(int status, int errnum, const char *format, ...);

14.4.13 execinfo.h

extern int backtrace(void **__array, int __size);
extern char **backtrace_symbols(void *const *__array, int __size);
extern void backtrace_symbols_fd(void *const *__array, int __size,
                               int __fd);

14.4.14 fcntl.h

#define AT_FDCWD -100 /* Use the current working directory to determine the target of relative file paths. */
#define POSIX_FADV_NORMAL 0
#define O_RDONLY 00
#define O_ACCMODE 0003
#define O_WRONLY 01
#define O_CREAT 0100
#define O_TRUNC 01000
#define O_DSYNC 010000
#define O_RDWR 02
#define O_EXCL 0200
#define O_APPEND 02000
#define O_SYNC 04010000
#define O_NOCTTY 0400
#define O_NDELAY 04000
#define O_NONBLOCK 04000
#define O_ASYNC 040000
#define O_CLOEXEC 1
#define AT_SYMLINK_NOFOLLOW 0x100 /* Do not follow symbolic links. */
#define AT_EMPTY_PATH 0x1000 /* Allow empty relative pathname. */
#define AT_EACCESS 0x200 /* Check access using effective user and group ID. */
#define AT_REMOVEDIR 0x2000 /* Remove directory instead of file. */
#define AT_SYMLINK_FOLLOW 0x400 /* Follow symbolic link. */
#define AT_NO_AUTOMOUNT 0x800 /* Suppress terminal automount traversal. */
#define FD_CLOEXEC 1
#define POSIX_FADV_RANDOM 1
#define _POSIX_FADV_SEQUENTIAL 2
#define POSIX_FADV_WILLNEED 3
#define _POSIX_FADV_RANDOM 1
#define F_DUPFD_CLOEXEC 1030 /* Duplicate file descriptor with the close-on-exec flag FD_CLOEXEC set. */
#define POSIX_FADV_SEQUENTIAL 2
#define POSIX_FADV_WILLNEED 3
#define O_RSYNC O_SYNC

struct flock {
    short l_type;
    short l_whence;
    off_t l_start;
    off_t l_len;
    pid_t l_pid;
};
struct flock64 {
    short l_type;
    short l_whence;
    loff_t l_start;
    loff_t l_len;
    pid_t l_pid;
};
#define AT_FDCWD -100
#define AT_SYMLINK_NOFOLLOW 0x100
#define AT_EACCESS 0x200
#define AT_REMOVEDIR 0x200
#define AT_SYMLINK_FOLLOW 0x400
#define F_DUPFD 0
#define F_RDLCK 0
#ifndef SEEK_SET
#define SEEK_SET 0
#endif
#define F_GETFD 1
#define F_WRLCK 1
#ifndef SEEK_CUR
#define SEEK_CUR 1
#endif
#define F_GETSIG 1
#define F_GETSIG 11
#define F_SETFD 2
#define F_GETLK 5
#define AT_FDCWD -100
#define AT_SYMLINK_NOFOLLOW 0x100
#define AT_EACCESS 0x200
#define AT_REMOVEDIR 0x200
#define AT_SYMLINK_FOLLOW 0x400
#define AT_FDCWD -100
#define AT_SYMLINK_NOFOLLOW 0x100
#define AT_EACCESS 0x200
#define AT_REMOVEDIR 0x200
#define AT_SYMLINK_FOLLOW 0x400
extern int creat(const char *__file, mode_t __mode);
extern int creat64(const char *__file, mode_t __mode);
extern int fcntl(int __fd, int __cmd, ...);
extern int open(const char *__file, int __oflag, ...);
extern int open64(const char *__file, int __oflag, ...);
extern int openat(int __fd, const char *__file, int __oflag, ...);
extern int openat64(int __fd, const char *__file, int __oflag, ...);
extern int posit_fadvise(int __fd, off_t __offset, off_t __len, int __advise);
extern int posit_fadvise64(int __fd, off64_t __offset, off64_t __len,
int __advise);
extern int posix_fallocate(int __fd, off_t __offset, off_t __len);
extern int posix_fallocate64(int __fd, off64_t __offset, off64_t __len);

14.4.15 fmtmsg.h

#define MM_HARD 1 /* Source of the condition is hardware. */
#define MM_NRECOV 128 /* Non-recoverable error. */
#define MM_UTIL 16 /* Condition detected by utility. */
#define MM_SOFT 2 /* Source of the condition is software. */
#define MM_PRINT 256 /* Display message in standard error. */
#define MM_OPSYS 32 /* Condition detected by operating system. */
#define MM_FIRM 4 /* Source of the condition is firmware. */
#define MM_CONSOLE 512 /* Display message on system console. */
#define MM_RECOVER 64 /* Recoverable error. */
#define MM_APPL 8 /* Condition detected by application. */
#define MM_NOSEV 0 /* No severity level provided for the message. */
#define MM_HALT 1 /* Error causing application to halt. */
#define MM_ERROR 2 /* Application has encountered a non-fatal fault. */
#define MM_WARNING 3 /* Application has detected unusual non-error condition. */
#define MM_INFO 4 /* Informative message. */
#define MM_NULLACT ((char *) 0)
#define MM_NULLLBL ((char *) 0)
#define MM_NULLTAG ((char *) 0)
#define MM_NULLTXT ((char *) 0)
#define MM_NULLMC ((long int) 0)
#define MM_NULLSEV 0
#define MM_NOTOK -1 /* The function failed completely. */
#define MM_OK 0 /* The function succeeded. */
#define MM_NOMSG 1 /* The function was unable to generate a message on standard error, but otherwise succeeded. */
#define MM_NOCON 4 /* The function was unable to generate a console message, but otherwise succeeded. */

extern int fmtmsg(long int __classification, const char *__label, int __severity, const char *__text, const char *__action, const char *__tag);

14.4.16 fnmatch.h

#define FNM_CASEFOLD (1<<4)
#define FNM_FILE_NAME FNM_PATHNAME
#define FNM_PATHNAME (1<<0)
#define FNM_NOESCAPE (1<<1)
#define FNM_PERIOD (1<<2)
#define FNM_NOMATCH 1

extern int fnmatch(const char *__pattern, const char *__name, int __flags);

14.4.17 ftw.h

#define FTW_D FTW_D
#define FTW_DNR FTW_DNR
#define FTW_DP FTW_DP
#define FTW_F FTW_F
#define FTW_NS FTW_NS
#define FTW_SL FTW_SL
#define FTW_SLN FTW_SLN

enum {
    FTW_F,
    FTW_D,
    FTW_DNR,
    FTW_NS,
    FTW_SL,
    FTW_DP,
    FTW_SLN
};

enum {
    FTW_PHYS = 1,
    FTW_MOUNT = 2,
    FTW_CHDIR = 4,
    FTW_DEPTH = 8
};

struct FTW {
    int base;
    int level;
};

typedef int (*__ftw_func_t) (const char *__filename, 
                          const struct stat * __status, int __flag);

typedef int (*__ftw64_func_t) (const char *__filename, 
                               const struct stat64 * __status, 
                               int __flag);

typedef int (*__nftw_func_t) (const char *__filename, 
                              const struct stat * __status, int __flag, 
                              struct FTW * __info);

typedef int (*__nftw64_func_t) (const char *__filename, 
                               const struct stat64 * __status, int __flag, 
                               struct FTW * __info);

extern int ftw(const char *__dir, __ftw_func_t __func, int __descriptors);

extern int ftw64(const char *__dir, __ftw64_func_t __func, int __descriptors);

extern int nftw(const char *__dir, __nftw_func_t __func, int __descriptors, int __flag);

extern int nftw64(const char *__dir, __nftw64_func_t __func, int __descriptors, int __flag);
### 14.4.18 getopt.h

```c
#define no_argument     0
#define required_argument       1
#define optional_argument       2

struct option {
    const char *name;
    int has_arg;
    int *flag;
    int val;
};

extern int getopt_long(int ___argc, char *const ___argv[],
const char *__shortopts,
const struct option *__longopts, int __longind);
extern int getopt_long_only(int ___argc, char *const ___argv[],
const char *__shortopts,
const struct option *__longopts, int __longind);
```

### 14.4.19 glob.h

```c
#define GLOB_ERR        (1<<0)
#define GLOB_MARK       (1<<1)
#define GLOB_BRACE      (1<<10)
#define GLOB_NOMAGIC    (1<<11)
#define GLOB_TILDE      (1<<12)
#define GLOB_ONLYDIR    (1<<13)
#define GLOB_TILDE_CHECK        (1<<14)
#define GLOB_NOSORT     (1<<2)
#define GLOB_DOOFFS     (1<<3)
#define GLOB_NOCHECK    (1<<4)
#define GLOB_APPEND     (1<<5)
#define GLOB_NOESCAPE   (1<<6)
#define GLOB_PERIOD     (1<<7)
#define GLOB_MAGCHAR    (1<<8)
#define GLOB_ALTDIRFUNC (1<<9)
#define GLOB_NOSPACE    1
#define GLOB_ABORTED    2
#define GLOB_NOMATCH    3
#define GLOB_NOSYS      4

typedef struct {
    size_t gl_pathc;
    char **gl_pathv;
    size_t gl_offs;
    int gl_flags;
    void (*gl_closedir) (void *);
    struct dirent (*gl_readdir) (void *);
    void (*gl_opendir) (const char *);
    int (*gl_lstat) (const char *, struct stat *);
    int (*gl_stat) (const char *, struct stat *);
} glob_t;

typedef struct {
    size_t gl_pathc;
    char **gl_pathv;
    size_t gl_offs;
    int gl_flags;
    void (*gl_closedir) (void *);
    struct dirent64 (*gl_readdir) (void *);
```
void *("gl_opendir") (const char *);
int (*"gl_lstat") (const char *, struct stat *);
int (*"gl_stat") (const char *, struct stat *);
} glob64_t;
extern int glob(const char *__pattern, int *__flags,
int (*__errfunc) (const char *, int), glob_t *__pglob);
extern int glob64(const char *__pattern, int *__flags,
int (*__errfunc) (const char *, int),
glob64_t *__pglob);
extern void globfree(glob_t *__pglob);
extern void globfree64(glob64_t *__pglob);

14.4.20 gnu/libc-version.h
extern const char *gnu_get_libc_release(void);
extern const char *gnu_get_libc_version(void);

14.4.21 grp.h
struct group {
    char *gr_name;
    char *gr_passwd;
    gid_t gr_gid;
    char **gr_mem;
};
extern void endgrent(void);
extern struct group *getgrent(void);
extern int getgrent_r(struct group *__resultbuf, char *__buffer,
size_t __buflen, struct group **__result);
extern struct group *getgrgid(gid_t __gid);
extern int getgrgid_r(gid_t __gid, struct group *__resultbuf,
char *__buffer, size_t __buflen,
struct group **__result);
extern struct group *getgrnam(const char *__name);
extern int getgrnam_r(const char *__name, struct group *
__resultbuf,
    char *__buffer, size_t __buflen,
    struct group **__result);
extern int getgrouplist(const char *__user, gid_t __group,
gid_t *__groups, int *__ngroups);
extern int initgroups(const char *__user, gid_t __group);
extern void setgrent(void);
extern int setgroups(size_t __n, const gid_t *__groups);

14.4.22 iconv.h
typedef void *iconv_t;
extern size_t iconv(iconv_t __cd, char **__inbuf, size_t * __inbytesleft,
    char **__outbuf, size_t * __outbytesleft);
extern int iconv_close(iconv_t __cd);
extern iconv_t iconv_open(const char *__tocode, const char *
_fromcode);

14.4.23 ifaddrs.h
#define ifa_broadaddr ifa_ifu.ifu_broadaddr
#define ifa_dstaddr ifa_ifu.ifu_dstaddr

© 2015 Linux Foundation
struct ifaddrs {
    struct ifaddrs *ifa_next;
    char *ifa_name;
    unsigned int ifa_flags;
    struct sockaddr *ifa_addr;
    struct sockaddr *ifa_netmask;
    union {
        struct sockaddr *ifu_broadaddr;
        struct sockaddr *ifu_dstaddr;
    } ifa_ifu;
    void *ifa_data;
};

extern void freeifaddrs(struct ifaddrs *);
extern int getifaddrs(struct ifaddrs **);

14.4.24 inttypes.h

#if !defined __cplusplus || defined __STDC_FORMAT_MACROS
#define PRId16  "d"
#define PRId32  "d"
#define PRId8   "d"
#define PRIdFAST8       "d"
#define PRIdLEAST16     "d"
#define PRIdLEAST32     "d"
#define PRIdLEAST8      "d"
#define SCNd32  "d"
#define SCNdLEAST32     "d"
#define SCNd16 "hd"
#define SCNdLEAST16     "hd"
#define SCNd8 "hdd"
#define SCNdFAST8       "hdd"
#define SCNd16 "hhi"
#define SCNdLEAST16     "hhi"
#define SCNd8 "hho"
#define SCNdFAST8 "hho"
#define SCNdLEAST16     "hho"
#define SCNd8 "hhu"
#define SCNdFAST8 "hhu"
#define SCNdLEAST16     "hhu"
#define SCNd8 "hhx"
#define SCNdFAST8 "hhx"
#define SCNdLEAST16     "hhx"
#define SCNi16  "hi"
#define SCNiLEAST16     "hi"
#define SCNo16  "ho"
#define SCNoLEAST16     "ho"
#define SCNu16  "hu"
#define SCNuLEAST16     "hu"
#define SCNx16 "hx"
#define SCNxFAST8 "hx"
#define SCNxLEAST16     "hx"
#define PRIi16  "i"
#define PRIiLEAST16     "i"
#define PRIi8   "i"
#define PRIiFAST8       "i"
#define PRIiLEAST32     "i"
#define PRIiLEAST8      "i"
#define PRIi8   "i"
#define PRIiFAST8       "i"
#define PRIiLEAST8      "i"
#define PRIi16 "o"
#define PRIi32 "o"
#define PRIi8   "o"
#endif
#define PRIoFAST8       "o"
#define PRIoLEAST16     "o"
#define PRIoLEAST32     "o"
#define PRIoLEAST8      "o"
#define SCNo32  "o"
#define SCNoLEAST32     "o"
#define PRIu16  "u"
#define PRIu32  "u"
#define PRIu8   "u"
#define PRIuFAST8       "u"
#define PRIuLEAST16     "u"
#define PRIuLEAST32     "u"
#define PRIuLEAST8      "u"
#define SCNu32  "u"
#define SCNuLEAST32     "u"
#define PRIX16  "X"
#define PRIX32  "X"
#define PRIX8   "X"
#define PRIXFAST8       "X"
#define PRIXLEAST16     "X"
#define PRIXLEAST32     "X"
#define PRIXLEAST8      "X"
#define SCNx32  "x"
#define SCNxLEAST32     "x"
#define PRId64  __PRI64_PREFIX"d"
#define PRIdFAST64      __PRI64_PREFIX"d"
#define PRIdLEAST64     __PRI64_PREFIX"d"
#define PRIdMAX __PRI64_PREFIX"d"
#define SCNd64  __PRI64_PREFIX"d"
#define SCNdFAST64      __PRI64_PREFIX"d"
#define SCNdLEAST64     __PRI64_PREFIX"d"
#define SCNdMAX __PRI64_PREFIX"d"
#define PRIi64  __PRI64_PREFIX"i"
#define PRIiFAST64      __PRI64_PREFIX"i"
#define PRIiLEAST64     __PRI64_PREFIX"i"
#define PRIiMAX __PRI64_PREFIX"i"
#define SCNi64  __PRI64_PREFIX"i"
#define SCNiFAST64      __PRI64_PREFIX"i"
#define SCNiLEAST64     __PRI64_PREFIX"i"
#define SCNiMAX __PRI64_PREFIX"i"
#define PRIo64  __PRI64_PREFIX"o"
#define PRIoFAST64      __PRI64_PREFIX"o"
#define PRIoLEAST64     __PRI64_PREFIX"o"
#define PRIoMAX __PRI64_PREFIX"o"
#define SCNoFAST64      __PRI64_PREFIX"o"
#define SCNoLEAST64     __PRI64_PREFIX"o"
#define SCNoMAX __PRI64_PREFIX"o"
#define PRIu64  __PRI64_PREFIX"u"
#define PRIuFAST64      __PRI64_PREFIX"u"
#define PRIuLEAST64     __PRI64_PREFIX"u"
#define PRIuMAX __PRI64_PREFIX"u"
#define SCNuFAST64      __PRI64_PREFIX"u"
#define SCNuLEAST64     __PRI64_PREFIX"u"
#define SCNuMAX __PRI64_PREFIX"u"
#define PRIX64  __PRI64_PREFIX"X"
#define PRIXFAST64      __PRI64_PREFIX"X"
#define PRIXLEAST64     __PRI64_PREFIX"X"
#define PRIxMAX __PRI64_PREFIX"x"
#define PRIx64 __PRI64_PREFIX"x"
#define PRIxFAST64 __PRI64_PREFIX"x"
#define PRIxLEAST64 __PRI64_PREFIX"x"
#define PRIxMAX __PRI64_PREFIX"x"
#define SCNx64 __PRI64_PREFIX"x"
#define SCNxFAST64 __PRI64_PREFIX"x"
#define SCNxLEAST64 __PRI64_PREFIX"x"
#define SCNxMAX __PRI64_PREFIX"x"
#define PRIdFAST16 __PRIPTR_PREFIX"d"
#define PRIdFAST32 __PRIPTR_PREFIX"d"
#define PRIdPTR __PRIPTR_PREFIX"d"
#define SCNdFAST16 __PRIPTR_PREFIX"d"
#define SCNdFAST32 __PRIPTR_PREFIX"d"
#define SCNdPTR __PRIPTR_PREFIX"d"
#define PRIiFAST16 __PRIPTR_PREFIX"i"
#define PRIiFAST32 __PRIPTR_PREFIX"i"
#define PRIiPTR __PRIPTR_PREFIX"i"
#define SCNiFAST16 __PRIPTR_PREFIX"i"
#define SCNiFAST32 __PRIPTR_PREFIX"i"
#define SCNiPTR __PRIPTR_PREFIX"i"
#define PRIoFAST16 __PRIPTR_PREFIX"o"
#define PRIoFAST32 __PRIPTR_PREFIX"o"
#define PRIoPTR __PRIPTR_PREFIX"o"
#define SCNoFAST16 __PRIPTR_PREFIX"o"
#define SCNoFAST32 __PRIPTR_PREFIX"o"
#define SCNoPTR __PRIPTR_PREFIX"o"
#define PRIuFAST16 __PRIPTR_PREFIX"u"
#define PRIuFAST32 __PRIPTR_PREFIX"u"
#define PRIuPTR __PRIPTR_PREFIX"u"
#define SCNuFAST16 __PRIPTR_PREFIX"u"
#define SCNuFAST32 __PRIPTR_PREFIX"u"
#define SCNuPTR __PRIPTR_PREFIX"u"
#define PRIXFAST16 __PRIPTR_PREFIX"X"
#define PRIXFAST32 __PRIPTR_PREFIX"X"
#define PRIXPTR __PRIPTR_PREFIX"X"
#define PRIxFAST16 __PRIPTR_PREFIX"x"
#define PRIxFAST32 __PRIPTR_PREFIX"x"
#define PRIxPTR __PRIPTR_PREFIX"x"
#define SCNxFAST16 __PRIPTR_PREFIX"x"
#define SCNxFAST32 __PRIPTR_PREFIX"x"
#define SCNxPTR __PRIPTR_PREFIX"x"
#endif

#define __PDP_ENDIAN 3412
#define PDP_ENDIAN __PDP_ENDIAN

extern intmax_t imaxabs(intmax_t __n);
extern imaxdiv_t imaxdiv(intmax_t __numer, intmax_t __denom);
extern intmax_t strtoimax(const char *__nptr, char **__endptr, int __base);
extern uintmax_t strtoumax(const char *__nptr, char **__endptr, int __base);
extern intmax_t wcstoimax(const wchar_t * __nptr, wchar_t * __endptr, int __base);
extern uintmax_t wcstoumax(const wchar_t * __nptr, wchar_t * __endptr, int __base);

14.4.25 langinfo.h

#define ABDAY_1 0x20000 /* Sun. */
#define ABDAY_2 0x20001
#define ABDAY_3 0x20002
#define ABDAY_4 0x20003
#define ABDAY_5 0x20004
#define ABDAY_6 0x20005
#define ABDAY_7 0x20006

#define DAY_1   0x20007
#define DAY_2   0x20008
#define DAY_3   0x20009
#define DAY_4   0x2000A
#define DAY_5   0x2000B
#define DAY_6   0x2000C
#define DAY_7   0x2000D

#define ABMON_1 0x2000E
#define ABMON_2 0x2000F
#define ABMON_3 0x20010
#define ABMON_4 0x20011
#define ABMON_5 0x20012
#define ABMON_6 0x20013
#define ABMON_7 0x20014
#define ABMON_8 0x20015
#define ABMON_9 0x20016
#define ABMON_10 0x20017
#define ABMON_11 0x20018
#define ABMON_12 0x20019

#define MON_1   0x2001A
#define MON_2   0x2001B
#define MON_3   0x2001C
#define MON_4   0x2001D
#define MON_5   0x2001E
#define MON_6   0x2001F
#define MON_7   0x20020
#define MON_8   0x20021
#define MON_9   0x20022
#define MON_10  0x20023
#define MON_11  0x20024
#define MON_12  0x20025

#define AM_STR  0x20026
#define PM_STR  0x20027

#define D_T_FMT 0x20028
#define D_FMT   0x20029
#define T_FMT   0x2002A
#define T_FMT_AMPM 0x2002B

#define ERA     0x2002C
#define ERA_D_FMT 0x2002E
#define ALT_DIGITS 0x2002F
#define ERA_D_T_FMT 0x20030
#define ERA_T_FMT 0x20031

#define CODESET 14

#define CRNCYSTR 0x4000F
#define RADIXCHAR 0x10000

extern char *nl_langinfo(nl_item __item);
14.4.26 libgen.h

#define basename __xpg_basename

extern char *__xpg_basename(char *__path);
extern char *dirname(char *__path);

14.4.27 libintl.h

extern char *bind_textdomain_codeset(const char *__domainname,
                                       const char *__codeset);
extern char *bindtextdomain(const char *__domainname,
                             const char *__dirname);
extern char *dcgettext(const char *__domainname, const char *
msgid,
           int __category);
extern char *dgettext(const char *__domainname, const char *
msgid);
extern char *dgettext(const char *__domainname, const char *
msgid1, const char *__msgid2, unsigned long int
__n, int __category);
extern char *dngettext(const char *__domainname, const char *
msgid1, const char *__msgid2, unsigned long int
__n);
extern char *gettext(const char *__msgid);
extern char *ngettext(const char *__msgid1, const char *__msgid2,
                      unsigned long int __n);
extern char *textdomain(const char *__domainname);

14.4.28 limits.h

#define LLONG_MIN       (-LLONG_MAX-1LL)
#define _POSIX_AIO_MAX  1
#define _POSIX_QLIMIT   1
#define _POSIX2_BC_STRING_MAX   1000
#define IOV_MAX 1024
#define _POSIX2_CHARCLASS_NAME_MAX      1
#define _POSIX_NAME_MAX 14
#define _POSIX2_UIO_MAXIOV 16
#define ULLONG_MAX      18446744073709551615ULL
#define _POSIX2_COLL_WEIGHTS_MAX        2
#define _POSIX_AIO_LISTIO_MAX   2
#define _POSIX_OPEN_MAX 20
#define _POSIX_CLOCKRES_MIN     20000000
#define CHARCLASS_NAME_MAX 2048
#define LINE_MAX 2048
#define _POSIX2_BC_DIM_MAX 2048
#define _POSIX2_LINE_MAX 2048
#define _POSIX_CHILD_MAX 25
#define COLL_WEIGHTS_MAX 255
#define NAME_MAX 255
#define _POSIX2_RE_DUP_MAX 255
#define _POSIX_HOST_NAME_MAX 255
#define _POSIX_MAX_CANON 255
#define _POSIX_MAX_INPUT 255
#define _POSIX_SYMLINK_MAX 255
#define _POSIX_PATH_MAX 256
#define _POSIX2_MAX_NSEMS_MAX 256
#define NGROUPS_MAX    32
#define WORD_BIT       32
#define _POSIX2_EXPR_NEST_MAX 32
#define _POSIX_DELAYTIMER_MAX 32
#define _POSIX_MQ_PRIO_MAX    32
#define _POSIX_SIGQUEUE_MAX   32
#define _POSIX_TIMER_MAX     32
#define _POSIX_SEM_VALUE_MAX 32767
#define _POSIX_SSIZE_MAX     32767
#define PATH_MAX          4096
#define _POSIX_ARG_MAX     4096
#define _POSIX_PIPE_BUF    512
#define _POSIX_TZNAME_MAX  8
#define _POSIX_LINK_MAX    8
#define _POSIX_MQ_OPEN_MAX 8
#define _POSIX_NGROUPS_MAX 8
#define _POSIX_RTSIG_MAX   8
#define _POSIX_STREAM_MAX  8
#define _POSIX_SYMLOOP_MAX 8
#define _POSIX_LOGIN_NAME_MAX 9
#define _POSIX_TTY_NAME_MAX 9
#define LLONG_MAX         9223372036854775807LL
#define _POSIX2_BC_BASE_MAX 99
#define _POSIX2_BC_SCALE_MAX 99
#define NL_MSGMAX         INT_MAX
#define NL_SETMAX         INT_MAX
#define NL_TEXTMAX        INT_MAX
#define SSIZE_MAX         LONG_MAX /* Maximum value of an object of type ssize_t */
#define BC_BASE_MAX       _POSIX2_BC_BASE_MAX
#define BC_DIM_MAX        _POSIX2_BC_DIM_MAX
#define BC_SCALE_MAX      _POSIX2_BC_SCALE_MAX
#define BC_STRING_MAX     _POSIX2_BC_STRING_MAX
#define EXPR_NEST_MAX     _POSIX2_EXPR_NEST_MAX
#define NL_LANGMAX        _POSIX2_LINE_MAX
#define NL_ARGMAX         _POSIX_ARG_MAX
#define __POSIX2_BC_BASE_MAX _POSIX2_BC_BASE_MAX
#define __POSIX2_BC_DIM_MAX _POSIX2_BC_DIM_MAX
#define __POSIX2_BC_SCALE_MAX _POSIX2_BC_SCALE_MAX
#define __POSIX2_BC_STRING_MAX _POSIX2_BC_STRING_MAX
#define __POSIX2_EXPR_NEST_MAX _POSIX2_EXPR_NEST_MAX
#define __POSIX2_LINE_MAX _POSIX2_LINE_MAX
#define MB_LEN_MAX        16
#define SCHAR_MIN         (-128)
#define SCHAR_MAX         127
#define UCHAR_MAX         255
#define CHAR_BIT          8
#define SHRT_MIN          (-32768)
#define SHRT_MAX          32767
#define USHRT_MAX         65535
#define INT_MIN           (-INT_MAX-1)
#define INT_MAX           2147483647
#define UINT_MAX          4294967295U
#define LONG_MIN          (-LONG_MAX-1L)
#define LONG_MAX          (LONG_MAX-1L)
#define PTHREAD_KEYS_MAX  1024
#define PTHREAD_THREADS_MAX 16384
#define PTHREAD_DESTRUCTOR_ITERATIONS 4

14.4.29 link.h

extern int
dl_iterate_phdr(int (*callback) (struct dl_phdr_info *, size_t, void *));
struct lconv {
    char *decimal_point;
    char *thousands_sep;
    char *grouping;
    char *int_curr_symbol;
    char *currency_symbol;
    char *mon_decimal_point;
    char *mon_thousands_sep;
    char *mon_grouping;
    char *positive_sign;
    char *negative_sign;
    char int_frac_digits;
    char frac_digits;
    char p_cs_precedes;
    char p_sep_by_space;
    char n_cs_precedes;
    char n_sep_by_space;
    char p_sign_posn;
    char n_sign_posn;
    char int_p_cs_precedes;
    char int_p_sep_by_space;
    char int_n_cs_precedes;
    char int_n_sep_by_space;
    char int_p_sign_posn;
    char int_n_sign_posn;
};

#define LC_GLOBAL_LOCALE        ((locale_t) -1L)
#define LC_CTYPE        0
#define LC_NUMERIC      1
#define LC_TELEPHONE    10
#define LC_MEASUREMENT  11
#define LC_IDENTIFICATION       12
#define LC_TIME 2
#define LC_COLLATE      3
#define LC_MONETARY     4
#define LC_MESSAGES     5
#define LC_ALL  6
#define LC_PAPER        7
#define LC_NAME 8
#define LC_ADDRESS      9

struct __locale_struct {
    struct locale_data *__locales[13];
    const unsigned short *__ctype_b;
    const int *__ctype_tolower;
    const int *__ctype_toupper;
    const char *__names[13];
};
typedef struct __locale_struct *__locale_t;
typedef struct __locale_struct *locale_t;

#define LC_ADDRESS_MASK (1 << LC_ADDRESS)
#define LC_COLLATE_MASK (1 << LC_COLLATE)
#define LC_IDENTIFICATION_MASK  (1 << LC_IDENTIFICATION)
#define LC_MEASUREMENT_MASK     (1 << LC_MEASUREMENT)
#define LC_MESSAGES_MASK        (1 << LC_MESSAGES)
#define LC_MONETARY_MASK        (1 << LC_MONETARY)
#define LC_NAME_MASK    (1 << LC_NAME)
#define LC_NUMERIC_MASK (1 << LC_NUMERIC)
#define LC_PAPER_MASK   (1 << LC_PAPER)
#define LC_TELEPHONE_MASK       (1 << LC_TELEPHONE)
#define LC_TIME_MASK    (1 << LC_TIME)
#define LC_CTYPE_MASK   (1<<LC_CTYPE)
#define LC_ALL_MASK     \
    (LC_CTYPE_MASK|  LC_NUMERIC_MASK|  LC_TIME_MASK|  \
    LC_COLLATE_MASK| LC_MONETARY_MASK|  \
    LC_MESSAGES_MASK|  LC_PAPER_MASK|  LC_NAME_MASK|  \
    LC_ADDRESS_MASK| LC_TELEPHONE_MASK|  \
    LC_MEASUREMENT_MASK| LC_IDENTIFICATION_MASK)

extern locale_t duplocale(locale_t __dataset);
extern void freelocale(locale_t __dataset);
extern struct lconv *localeconv(void);
extern  locale_t  newlocale(int  __category_mask, const  char *
locale, __locale, __base);
extern char *setlocale(int __category, const char *__locale);
extern locale_t uselocale(locale_t __dataset);

14.4.31 lsb/time.h

struct timeval {
    time_t tv_sec;
    suseconds_t tv_usec;
};

14.4.32 lsb/types.h

/*
 * This header is architecture dependent
 * Please refer to the specific architecture specification for details
 */

14.4.33 lsb/wchar.h

typedef unsigned int wint_t;
typedef struct {
    int count;
    wint_t value;
} __mbstate_t;
typedef __mbstate_t mbstate_t;

14.4.34 monetary.h

extern ssize_t strfmon(char *__s, size_t __maxsize, const char *
format, ...);
extern ssize_t strfmon_l(char *s, size_t maxsize, locale_t
locale, const char *format, ...);

14.4.35 net/if.h

#define IF_NAMESIZE     16
#define IFF_UP 0x01 /* Interface is up. */
#define IFF_BROADCAST 0x02 /* Broadcast address valid. */
#define IFF_DEBUG 0x04 /* Turn on debugging. */
#define IFF_LOOPBACK 0x08 /* Is a loopback net. */
#define IFF_POINTOPOINT 0x10 /* Interface is point-to-point link. */
#define IFF_PROMISC 0x100 /* Receive all packets. */
#define IFF_MULTICAST 0x1000 /* Supports multicast. */
#define IFF_NOTRAILERS 0x20 /* Avoid use of trailers. */
#define IFF_RUNNING 0x40 /* Resources allocated. */
#define IFF_NOARP 0x80 /* No address resolution protocol. */

struct if_nameindex {
    unsigned int if_index; /* 1, 2, ... */
    char *if_name; /* null terminated name: */
};

struct ifaddr {
    struct sockaddr ifa_addr; /* Address of interface. */
    union {
        struct sockaddr ifu_broadaddr;
        struct sockaddr ifu_dstaddr;
    } ifa_ifu;
    void *ifa_ifp;
    void *ifa_next;
};

#define ifr_name        ifr_ifrn.ifrn_name /* interface name */
#define ifr_addr        ifr_ifru.ifru_addr /* address */
#define ifr_broadaddr    ifr_ifru.ifru_broadaddr /* broadcast address */
#define ifr_data        ifr_ifru.ifru_data /* for use by interface */
#define ifr_dstaddr     ifr_ifru.ifru_dstaddr /* other end of p-p link */
#define ifr_flags       ifr_ifru.ifru_flags /* flags */
#define ifr_hwaddr      ifr_ifru.ifru_hwaddr /* interface name */
#define ifr_bandwidth   ifr_ifru.ifru_ivalue /* link bandwidth */
#define ifr_ifindex     ifr_ifru.ifru_ivalue /* interface index */
#define ifr_metric      ifr_ifru.ifru_ivalue /* metric */
#define ifr_qlen        ifr_ifru.ifru_ivalue /* queue length */
#define ifr_mtu         ifr_ifru.ifru_mtu /* mtu */
#define ifr_netmask     ifr_ifru.ifru_netmask /* interface net mask */
#define ifr_slave       ifr_ifru.ifru_slave /* slave device */
#define IFNAMSIZ        IF_NAMESIZE

struct ifreq {
    union {
        char ifrn_name[IFNAMSIZ];
    } ifr_ifrn;
    union {
        struct sockaddr ifru_addr;
        struct sockaddr ifru_dstaddr;
        struct sockaddr ifru_broadaddr;
        struct sockaddr ifru_netmask;
        struct sockaddr ifru_hwaddr;
        short ifru_flags;
    } ifr_ifru;
}
struct ifconf {
    int ifc_len;
    union {
        caddr_t ifcu_buf;
        struct ifreq *ifcu_req;
    } ifc_ifcu;
};

#define ifc_buf ifc_ifcu.ifcu_buf       /* Buffer address. */
#define ifc_req ifc_ifcu.ifcu_req       /* Array of structures. */

struct hostent {
    char *h_name;
    char **h_aliases;
    int h_addrtype;
    int h_length;
    char **h_addr_list;
};

struct servent {
    char *s_name;
    char **s_aliases;
    int s_port;
    char *s_proto;
};

struct protoent {
    char *p_name;
    char **p_aliases;
    int p_proto;
};

struct netent {
    char *n_name;
};

14.4.36 netdb.h

#define h_errno (*__h_errno_location ())
#define NETDB_INTERNAL  -1       /* See errno. */
#define NETDB_SUCCESS   0       /* No problem. */
#define HOST_NOT_FOUND  1       /* Authoritative Answer Host not found. */
#define IPPORT_RESERVED 1024
#define NI_MAXHOST      1025
#define TRY_AGAIN       2       /* Non-Authoritative Host not found, or SERVFAIL. */
#define NO_RECOVERY     3       /* Non recoverable errors, FORMERR, REFUSED, NOTIMP. */
#define NI_MAXSERV      32
#define NO_DATA 4               /* Valid name, no data record of requested type. */
#define h_addr  h_addr_list[0]
#define NO_ADDRESS      NO_DATA /* No address, look for MX record. */

struct servent {
    char *s_name;
    char **s_aliases;
    int s_port;
    char *s_proto;
};

struct hostent {
    char *h_name;
    char **h_aliases;
    int h_addrtype;
    int h_length;
    char **h_addr_list;
};

struct protoent {
    char *p_name;
    char **p_aliases;
    int p_proto;
};

struct netent {
    char *n_name;
}
char **n_aliases;
int n_addrtype;
unsigned int n_net;
};

#define AI_PASSIVE 0x0001 /* Socket address is intended for `bind' */
#define AI_CANONNAME 0x0002 /* Request for canonical name */
#define AI_NUMERICHOST 0x0004 /* Don't use name resolution */
#define AI_V4MAPPED 0x0008 /* IPv4 mapped addresses are acceptable. */
#define AI_ALL 0x0010 /* Return IPv4 mapped and IPv6 addresses. */
#define AI_ADDRCONFIG 0x0020 /* Use configuration of this host to choose returned address type. */
#define AI_NUMERICSERV 0x0400 /* Don't use name resolution */

struct addrinfo {
    int ai_flags;
    int ai_family;
    int ai_socktype;
    int ai_protocol;
    socklen_t ai_addrlen;
    struct sockaddr *ai_addr;
    char *ai_canonname;
    struct addrinfo *ai_next;
};

#define NI_NUMERICHOST 1
#define NI_DGRAM 16
#define NI_NUMERICSERV 2
#define NI_NOFQDN 4
#define NI_NAMEREQD 8
#define EAI_BADFLAGS -1 /* Invalid value for `ai_flags' field. */
#define EAI_MEMORY -10 /* Memory allocation failure. */
#define EAI_SYSTEM -11 /* System error returned in `errno'. */
#define EAI_NONAME -2 /* NAME or SERVICE is unknown. */
#define EAI_AGAIN -3 /* Temporary failure in name resolution. */
#define EAI_FAIL -4 /* Non-recoverable failure in name res. */
#define EAI_NODATA -5 /* No address associated with NAME. */
#define EAI_FAMILY -6 /* `ai_family' not supported. */
#define EAI_SOCKTYPE -7 /* `ai_family' not supported. */
#define EAI_SERVICE -8 /* SERVICE not supported for `ai_socktype'. */
#define EAI_ADDRFAMILY -9 /* Address family for NAME not supported. */

extern int *__h_errno_location(void);
extern void endprotoent(void);
extern void endservent(void);
extern void freeaddrinfo(struct addrinfo *__ai);
extern const char *gai_strerror(int __ecode);
extern int getaddrinfo(const char *__name, const char *__service, const struct addrinfo *__req, struct addrinfo **__pai);
extern struct hostent *gethostbyaddr(const void *__addr, socklen_t __len, int __type);
extern int gethostbyaddr_r(const void *__addr, socklen_t __len, int __type,
14.4.37 netinet/icmp6.h

#define ICMP6_FILTER_WILLBLOCK(type, filterp) (((((filterp) - >icmp6_filt[(type) >> 5]) & (1 << ((type) & 31))) != 0)
#define ICMP6_FILTER_WILLPASS(type, filterp) (((((filterp) - >icmp6_filt[(type) >> 5]) & (1 << ((type) & 31))) == 0)
#define ICMP6_FILTER_SETPASS(type, filterp) (((filterp) - icmp6_filter[(type) >> 5]) &= ~(1 << ((type) & 31)))
#define ICMP6_FILTER_SETBLOCK(type, filterp) (((filterp) - icmp6_filter[(type) >> 5]) |= (1 << ((type) & 31)))
#define ICMP6_DST_UNREACH_NOROUTE 0
#define ICMP6_PARAMPROB_HEADER 0
#define ICMP6_TIME_EXCEEDED_TRANSIT 0
#define ICMP6_RR_FLAGS_PREVDONE 0x08
#define ICMP6_RR_FLAGS_SPECSITE 0x10
#define ICMP6_RR_PCOUSE_RAFLAGS_AUTO 0x10
#define ICMP6_RR_PCOUSE_RAFLAGS_FORCEAPPLY 0x20
#define ICMP6_RR_PCOUSE_RAFLAGS_ONLINK 0x20
#define ND_OPT_PI_FLAG_RADDR 0x00
#define ND_RA_FLAG_HOME_AGENT 0x00
#define ICMP6_RR_FLAGS_REQRESULT 0x40
#define ND_OPT_PI_FLAG_AUTO 0x40
#define ND_RA_FLAG_OTHER 0x40
#define ICMP6_INFOMSG_MASK 0x80
#define ICMP6_RR_FLAGS_TEST 0x80
#define ND_OPT_PI_FLAG_ONLINK 0x80
#define ND_RA_FLAG_MANAGED 0x80
#define ICMP6_INFOMSG 1
#define ICMP6_ROUTER_RENUMBERING 138
#define ICMP6_ROUTER_ADVNUMBERING 2
#define ICMP6_PACKETINFO 2
#define ND_OPT_PI_FLAG_ARP 2
#define ICMP6_PCO_ADD 1
#define ICMP6_ECHO_REQUEST 128
#define ICMP6_ECHO_REPLY 129
#define MLD_LISTENER_QUERY 130
#define MLD_LISTENER_REPORT 131
#define MLD_LISTENER_REDUCTION 132
#define ND_ROUTER_SOLICIT 133
#define ND_ROUTER_ADVERT 134
#define ND_NEIGHBOR_SOLICIT 135
#define ND_NEIGHBOR_ADVERT 136
#define ND_REDIRECT 137
#define ICMP6_ROUTER_RENUMBERING 138
#define ICMP6_DST_UNREACH_BEYONDSCOPE 2
#define ICMP6_FILTER_PASS 2
#define ICMP6_PACKET_TOO_BIG 2
#define ICMP6_TIME_EXCEEDED 2
#define ND_OPT_PI_FLAG_HOME_AGENT_INFO 2
#define ICMP6_PCO_CHANGE 3
#define ICMP6_PCO_SETGLOBAL 3
#define ICMP6_DST_UNREACH_NOPORT 4
#define ICMP6_FILTER_PASSONLY 4
#define ICMP6_PARAM_PROB 4
#define ND_OPT_PI_FLAG_REDIRECTED_HEADER 4
#define ND_OPT_MTU 5
#define ND_OPT_RTR_ADV_INTERVAL 7
#define ND_OPT_HOME_AGENT_INFO 8
#define icmp6_id ICMP6_DATA16[0]
#define icmp6_maxdelay ICMP6_DATA16[0]
#define icmp6_seq ICMP6_DATA16[1]
#define icmp6_mtu ICMP6_DATA32[0]
#define icmp6_ptr ICMP6_DATA32[0]
#define icmp6_data16 ICMP6_DATA16[1]
#define icmp6_data32 ICMP6_DATA32[0]
#define icmp6_data8 ICMP6_DATA8
#define ICMP6_FILTER_SETPASSALL(filterp)        memset (filterp, 0, sizeof (struct icmp6_filter));
#define ICMP6_FILTER_SETBLOCKALL(filterp)       memset (filterp, 0xFF, sizeof (struct icmp6_filter));
#define mld_cksum       mld_icmp6_hdr.icmp6_cksum
#define mld_code        mld_icmp6_hdr.icmp6_code
#define mld_maxdelay    mld_icmp6_hdr.icmp6_data16[0]
#define mld_reserved    mld_icmp6_hdr.icmp6_data16[1]
#define mld_type        mld_icmp6_hdr.icmp6_type
#define nd_na_cksum     nd_na_hdr.icmp6_cksum
#define nd_na_code      nd_na_hdr.icmp6_code
#define nd_na_flags_reserved    nd_na_hdr.icmp6_data32[0]
#define nd_na_type      nd_na_hdr.icmp6_type
#define nd_ns_cksum     nd_ns_hdr.icmp6_cksum
#define nd_ns_code      nd_ns_hdr.icmp6_code
#define nd_ns_reserved  nd_ns_hdr.icmp6_data32[0]
#define nd_ns_type      nd_ns_hdr.icmp6_type
#define nd_ra_cksum     nd_ra_hdr.icmp6_cksum
#define nd_ra_code      nd_ra_hdr.icmp6_code
#define nd_ra_flags_reserved    nd_ra_hdr.icmp6_data32[0]
#define nd_ra_router_lifetime   nd_ra_hdr.icmp6_data16[1]
#define nd_ra_reachable nd_ra_hdr.icmp6_data8[0]
#define nd_ra_retransmit nd_ra_hdr.icmp6_data8[1]
#define nd_ra_type      nd_ra_hdr.icmp6_type
#define nd_rd_cksum     nd_rd_hdr.icmp6_cksum
#define nd_rd_code      nd_rd_hdr.icmp6_code
#define nd_rd_reserved  nd_rd_hdr.icmp6_data32[0]
#define nd_rd_type      nd_rd_hdr.icmp6_type
#define nd_rs_cksum     nd_rs_hdr.icmp6_cksum
#define nd_rs_code      nd_rs_hdr.icmp6_code
#define nd_rs_reserved  nd_rs_hdr.icmp6_data32[0]
#define nd_rs_type      nd_rs_hdr.icmp6_type
#define rr_codenum rr_hdr.icmp6_code
#define rr_seqnum       rr_hdr.icmp6_data32[0]
#define rr_type rr_hdr.icmp6_type

struct icmp6_filter {
    uint32_t icmp6_filt[8];
};

struct icmp6_hdr {
    uint8_t icmp6_type;
    uint8_t icmp6_code;
    uint16_t icmp6_cksum;
    union {
        uint32_t icmp6_un_data32[1];
        uint16_t icmp6_un_data16[2];
        uint8_t  icmp6_un_data8[4];
    } icmp6_dataun;
};

struct nd_router_solicit {
    struct icmp6_hdr nd_rs_hdr;
};

struct nd_router_advert {
    struct icmp6_hdr nd_ra_hdr;
    uint32_t nd_ra_reachable;
    uint32_t nd_ra_retransmit;
};

struct nd_neighbor_solicit {
    struct icmp6_hdr nd_ns_hdr;
    struct in6_addr nd_ns_target;
};

struct nd_neighbor_advert {
    struct icmp6_hdr nd_na_hdr;
    struct in6_addr nd_na_target;
};

struct nd_redirect {
struct icmp6_hdr nd_rd_hdr;
struct in6_addr nd_rd_target;
struct in6_addr nd_rd_dst;
}
struct nd_opt_hdr {
    uint8_t nd_opt_type;
    uint8_t nd_opt_len;
};
struct nd_opt_prefix_info {
    uint8_t nd_opt_pi_type;
    uint8_t nd_opt_pi_len;
    uint8_t nd_opt_pi_prefix_len;
    uint8_t nd_opt_pi_flags_reserved;
    uint32_t nd_opt_pi_valid_time;
    uint32_t nd_opt_pi_preferred_time;
    uint32_t nd_opt_pi_reserved2;
    struct in6_addr nd_opt_pi_prefix;
};
struct nd_opt_rd_hdr {
    uint8_t nd_opt_rh_type;
    uint8_t nd_opt_rh_len;
    uint16_t nd_opt_rh_reserved1;
    uint32_t nd_opt_rh_reserved2;
};
struct nd_opt_mtu {
    uint8_t nd_opt_mtu_type;
    uint8_t nd_opt_mtu_len;
    uint16_t nd_opt_mtu_reserved;
    uint32_t nd_opt_mtu_mtu;
};
struct mld_hdr {
    struct icmp6_hdr mld_icmp6_hdr;
    struct in6_addr mld_addr;
};
struct icmp6_router_renum {
    struct icmp6_hdr rr_hdr;
    uint8_t rr_segunum;
    uint8_t rr_flags;
    uint16_t rr_maxdelay;
    uint32_t rr_reserved;
};
struct rr_pco_match {
    uint8_t rpm_code;
    uint8_t rpm_len;
    uint8_t rpm_ordinal;
    uint8_t rpm_matchlen;
    uint8_t rpm_minlen;
    uint8_t rpm_maxlen;
    uint16_t rpm_reserved;
    struct in6_addr rpm_prefix;
};
struct rr_pco_use {
    uint8_t rpu_uselen;
    uint8_t rpu_keeplen;
    uint8_t rpu_ramask;
    uint8_t rpu_raflags;
    uint32_t rpu_vlttime;
    uint32_t rpu_pltime;
    uint32_t rpu_flags;
    struct in6_addr rpu_prefix;
};
struct rr_result {
    uint16_t rrr_flags;
    uint8_t rrr_ordinal;
    uint8_t rrr_matchedlen;
    uint32_t rrr_ifid;
#include <netinet/in.h>

struct in6_addr rrr_prefix;

struct nd_opt_adv_interval {
    uint8_t nd_opt_adv_interval_type;
    uint8_t nd_opt_adv_interval_len;
    uint16_t nd_opt_adv_interval_reserved;
    uint32_t nd_opt_adv_interval_ival;
};

struct nd_opt_home_agent_info {
    uint8_t nd_opt_home_agent_info_type;
    uint8_t nd_opt_home_agent_info_len;
    uint16_t nd_opt_home_agent_info_reserved;
    int16_t nd_opt_home_agent_info_preference;
    uint16_t nd_opt_home_agent_info_lifetime;
};

14.4.38 netinet/igmp.h

#define IGMP_MEMBERSHIP_QUERY   0x11
#define IGMP_V1_MEMBERSHIP_REPORT       0x12
#define IGMP_DVMRP      0x13
#define IGMP_PIM        0x14
#define IGMP_TRACE      0x15
#define IGMP_V2_MEMBERSHIP_REPORT       0x16
#define IGMP_V2_LEAVE_GROUP     0x17
#define IGMP_MTRACE_RESP        0x1e
#define IGMP_MTRACE      0x1f
#define IGMP_DELAYING_MEMBER    1
#define IGMP_v1_ROUTER  1
#define IGMP_MAX_HOST_REPORT_DELAY 10
#define IGMP_TIMER_SCALE        10
#define IGMP_IDLE_MEMBER        2
#define IGMP_v2_ROUTER  2
#define IGMP_LAZY_MEMBER        3
#define IGMP_SLEEPING_MEMBER    4
#define IGMP_AWAKENING_MEMBER   5
#define IGMP_MINLEN     8
#define IGMP_HOST_MEMBERSHIP_QUERY      IGMP_MEMBERSHIP_QUERY
#define IGMP_HOST_MEMBERSHIP_REPORT     IGMP_V1_MEMBERSHIP_REPORT
#define IGMP_HOST_LEAVE_MESSAGE IGMP_V2_LEAVE_GROUP
#define IGMP_HOST_NEW_MEMBERSHIP_REPORT IGMP_V2_MEMBERSHIP_REPORT

struct igmp {
    u_int8_t igmp_type;
    u_int8_t igmp_code;
    u_int16_t igmp_cksum;
    struct in_addr igmp_group;
};

14.4.39 netinet/in.h

#define IPPROTO_IP      0
#define IPPROTO_ICMP    1
#define IPPROTO_UDP     17
#define IPPROTO_IGMP    2
#define IPPROTO_RAW     255
#define IPPROTO_IPV6    41
#define IPPROTO_ICMPV6  58
#define IPPROTO_TCP     6

typedef uint16_t in_port_t;

struct in_addr {
typedef uint32_t in_addr_t;
#define INADDR_NONE     ((in_addr_t) 0xffffffff)
#define INADDR_BROADCAST        (0xffffffff)
#define INADDR_ANY      0
#define INADDR_LOOPBACK 0x7f000001      /* 127.0.0.1 */
#define s6_addr16 in6_u.u6_addr16
#define s6_addr32 in6_u.u6_addr32
#define s6_addr in6_u.u6_addr8

struct in6_addr {
  union {
    uint8_t u6_addr8[16];
    uint16_t u6_addr16[8];
    uint32_t u6_addr32[4];
  } in6_u;
};

#define IN6ADDR_ANY_INIT
{
  { { 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 } } } 
#define IN6ADDR_LOOPBACK_INIT
{ { 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1 } } 

#define IN_MULTICAST(a)  ((((in_addr_t) (a))&0xf0000000)==0xe0000000)
#define INET_ADDRSTRLEN 16

struct sockaddr_in {
  sa_family_t sin_family;
  unsigned short sin_port;
  struct in_addr sin_addr;
  unsigned char sin_zero[8];
};

#define IN6_IS_ADDR_LINKLOCAL(a)        ((((const uint32_t *) (a))[0] & htonl (0xffc00000)) == htonl (0xfe800000))
#define IN6_IS_ADDR_SITELOCAL(a)        ((((const uint32_t *) (a))[0] & htonl (0xffc00000)) == htonl (0xfec00000))
#define IN6_ARE_ADDR_EQUAL(a,b) ((((const uint32_t *) (a))[0] == ((const uint32_t *) (b))[0]) && (((const uint32_t *) (a))[1] == ((const uint32_t *) (b))[1]) && (((const uint32_t *) (a))[2] == ((const uint32_t *) (b))[2]) && (((const uint32_t *) (a))[3] == ((const uint32_t *) (b))[3]))
#define IN6_IS_ADDR_V4COMPAT(a) ((((const uint32_t *) (a))[0] == 0) && (((const uint32_t *) (a))[1] == 0) && (((const uint32_t *) (a))[2] == 0) && (ntohl (((const uint32_t *) (a))[3]) > 1))
#define IN6_IS_ADDR_V4MAPIED(a) ((((const uint32_t *) (a))[0] == 0) && (((const uint32_t *) (a))[1] == 0) && (((const uint32_t *) (a))[2] == htonl (0xffff)))
#define IN6_IS_ADDR_UNSPECIFIED(a)      (((const uint32_t *) (a))[0] == 0 && ((const uint32_t *) (a))[1] == 0 && ((const uint32_t *) (a))[2] == 0 && ((const uint32_t *) (a))[3] == 0)
#define IN6_IS_ADDR_MULTICAST(a)        ((((const uint8_t *) (a))[0] == 0xff))
#define IN6_IS_ADDR_MC_NODELOCAL(a)     (IN6_IS_ADDR_MULTICAST(a) && (((const uint8_t *) (a))[1] & 0xf) == 0x1)
#define IN6_IS_ADDR_MC_LINKLOCAL(a)     (IN6_IS_ADDR_MULTICAST(a) && (((const uint8_t *) (a))[1] & 0xf) == 0x2)
#define IN6_IS_ADDR_MC_SITELOCAL(a)     (IN6_IS_ADDR_MULTICAST(a) && (((const uint8_t *) (a))[1] & 0xf) == 0x5))
#define IN6_IS_ADDR_MC_ORGLOCAL(a) ((IN6_IS_ADDR_MULTICAST(a) && (((const uint8_t *) (a))[1] & 0xf) == 0x8))
#define IN6_IS_ADDR_MC_GLOBAL(a) ((IN6_IS_ADDR_MULTICAST(a) && (((const uint8_t *) (a))[1] & 0xf) == 0xe))
#define INET6_ADDRSTRLEN 46
struct sockaddr_in6 {
    unsigned short sin6_family; /* AF_INET6 */
    uint16_t sin6_port; /* Transport layer port */
    uint32_t sin6_flowinfo; /* IPv6 flow information */
    struct in6_addr sin6_addr; /* IPv6 address */
    uint32_t sin6_scope_id; /* scope id (new in RFC2553) */
};
#define SOL_IP 0
#define IP_TOS 1 /* IP type of service and precedence */
#define IPV6_UNICAST_HOPS 16
#define IPV6_MULTICAST_IF 17
#define IPV6_MULTICAST_HOPS 18
#define IPV6_MULTICAST_LOOP 19
#define IP_TTL 2 /* IP time to live */
#define IPV6_JOIN_GROUP 20
#define IPV6_LEAVE_GROUP 21
#define IPV6_V6ONLY 26
#define IPV6_MULTICAST_IF 32 /* set/get IPv6 multicast i/f */
#define IPV6_MULTICAST_TTL 33 /* set/get IPv6 multicast ttl */
#define IPV6_MULTICAST_LOOP 34 /* set/get IPv6 multicast loopback */
#define IP_ADD_MEMBERSHIP 35 /* add an IP group membership */
#define IP_DROP_MEMBERSHIP 36 /* drop an IP group membership */
#define IP_OPTIONS 4 /* IP per-packet options */
#define IPV6_ADD_MEMBERSHIP IPV6_JOIN_GROUP
#define IPV6_DROP_MEMBERSHIP IPV6_LEAVE_GROUP
struct ipv6_mreq {
    struct in6_addr ipv6mr_multiaddr; /* IPv6 multicast address of group */
    int ipv6mr_interface; /* local IPv6 address of interface */
};
struct ip_mreq {
    struct in_addr imr_multiaddr; /* IP multicast address of group */
    struct in_addr imr_interface; /* local IP address of interface */
};
extern int bindresvport(int, struct sockaddr_in *);
extern const struct in6_addr in6addr_any;
extern const struct in6_addr in6addr_loopback;

14.4.40 netinet/in_systm.h

typedef u_int16_t n_short;
typedef u_int32_t n_long;
typedef u_int32_t n_time;

14.4.41 netinet/ip.h

#define IPOPT_CLASS(o) (((o) & IPOPT_CLASS_MASK)
#define IPOPT_COPIED(o) (((o) & IPOPT_COPY))
#define IPOPT_NUMBER(o) (((o) & IPOPT_NUMBER_MASK))
#define IPOPT_EOL 0
#define IPOPT_OPTVAL 0
#define IPOPT_TS_TSONLY 0
#define IPOPT_CONTROL 0x00
#define IPOPT_SECUR_UNCLASS 0x0000
#define IPOPT_NUMBER_MASK 0x1f
#define IP_OFFMASK 0x1fff
#define IPOPT_RESERVED1 0x20
#define IP_MF 0x2000
#define IPOPT_DEBMEAS 0x40
#define IP_DF 0x4000
#define IPOPT_CLASS_MASK 0x60
#define IPOPT_RESERVED2 0x60
#define IPOPT_SECUR_TOPSECRET 0x6bc5
#define IPOPT_SECUR_EFTO 0x789a
#define IPOPT_COPY 0x80
#define IP_RF 0x8000
#define IPOPT_SECUR_RESTR 0xaf13
#define IPOPT_SECURITY 0xbc4d
#define IPOPT_CLASS 0xdf88
#define IPOPT_NOP 1
#define IPOPT_OLEN 1
#define IPTTLDEC 1
#define IPOPT_SECURITY 130
#define IPOPT_LSRR 131
#define IPOPT_SATID 136
#define IPOPT_SSRR 137
#define IPOPT_RA 148
#define IPOPT_OFFSET 2
#define MAXTTL 255
#define IPOPT_TS_PRESPEC 3
#define IPOPT_MINOFF 4
#define IPVERSION 4
#define MAX_IPOPTLEN 40
#define IP_MSS 576
#define IPFRAGTTL 60
#define IPDEFTTL 64
#define IP_MAXPACKET 65535
#define IPOPT_TS 68
#define IPOPT_RR 7
#define IPOPT_MEASUREMENT IPOPT_DEBMEAS
#define IPOPT_END IPOPT_EOL
#define IPOPT_NOOP IPOPT_NOP
#define IPOPT_SID IPOPT_SATID
#define IPOPT_SEC IPOPT_SECURITY
#define IPOPT_TIMESTAMP IPOPT_TS

#define IPTOS_TOS(tos) (((tos) & IPTOS_TOS_MASK))
#define IPTOS_LOWCAST 0x02
#define IPTOS_RELIABILITY 0x04
#define IPTOS_THROUGHPUT 0x08
#define IPTOS_LOWDELAY 0x10
#define IPTOS_TOS_MASK 0x1e
#define IPTOS_MINCOST IPTOS_LOWCAST

#define IPTOS_PREC(tos) (((tos) & IPTOS_PREC_MASK))
#define IPTOS_PREC_MASK 0xe0

14.4.42 netinet/ip.h

#define IP6OPT_TYPE(o) (((o) & 0xc0))
#define IP6OPT_PAD1 0
#define IP6OPT_TYPE_SKIP 0x00
#define IP6OPT_TUNNEL_LIMIT 0x04
#define IP6OPT_ROUTER_ALERT 0x05
#define IP6OPT_TYPE_MUTABLE 0x20
#define IP6OPT_TYPE_DISCARD 0x40
#define IP6OPT_TYPE_FORCEICMP 0x80
#define IP6OPT_TYPE_ICMP 0xc0
#define IP6OPT_JUMBO 0xc2
#define IP6OPT_NSAP_ADDR 0xc3
#define IP6OPT_PADN 1
#define IP6OPT_JUMBO_LEN 6
#define ip6_flow ip6_ctlun.ip6_un1.ip6_un1_flow
#define ip6_hlim ip6_ctlun.ip6_un1.ip6_un1_hlim
#define ip6_hops ip6_ctlun.ip6_un1.ip6_un1_hlim
#define ip6_nxt ip6_ctlun.ip6_un1.ip6_un1_nxt
#define ip6_plen ip6_ctlun.ip6_un1.ip6_un1_plen
#define ip6_vfc ip6_ctlun.ip6_un2_vfc

struct ip6_hdrctl {
    uint32_t ip6_un1_flow;
    uint16_t ip6_un1_plen;
    uint8_t ip6_un1_nxt;
    uint8_t ip6_un1_hlim;
};

struct ip6_hdr {
    struct in6_addr ip6_src;
    struct in6_addr ip6_dst;
};

struct ip6_ext {
    uint8_t ip6e_nxt;
    uint8_t ip6e_len;
};

struct ip6_hbh {
    uint8_t ip6h_nxt;
    uint8_t ip6h_len;
};

struct ip6_dest {
    uint8_t ip6d_nxt;
    uint8_t ip6d_len;
};

struct ip6_rthdr {
    uint8_t ip6r_nxt;
    uint8_t ip6r_len;
    uint8_t ip6r_type;
    uint8_t ip6r_segleft;
};

struct ip6_frag {
    uint8_t ip6f_nxt;
    uint8_t ip6f_reserved;
    uint16_t ip6f_offlg;
    uint32_t ip6f_ident;
};

struct ip6_opt {
    uint8_t ip6o_type;
    uint8_t ip6o_len;
};

struct ip6_opt_jumbo {
    uint8_t ip6oj_type;
    uint8_t ip6oj_len;
    uint8_t ip6oj_jumbo_len[4];
};

struct ip6_opt_nsap {
    uint8_t ip6on_type;
    uint8_t ip6on_len;
    uint8_t ip6on_src_nsap_len;

© 2015 Linux Foundation
14 Base Libraries

struct ip6_opt_tunnel {
    uint8_t ip6ot_type;
    uint8_t ip6ot_len;
    uint8_t ip6ot_encap_limit;
};

struct ip6_opt_router {
    uint8_t ip6or_type;
    uint8_t ip6or_len;
    uint8_t ip6or_value[2];
};

14.4.43 netinet/ip_icmp.h

#define ICMP_INFOTYPE(type)     
    ((type) == ICMP_ECHOREPLY || (type) == ICMP_ECHO || 
     (type) == ICMP_ROUTERADVVERT || (type) == ICMP_ROUTERADVERT || 
     (type) == ICMP_TSTAMP || (type) == ICMP_TSTAMPREPLY || 
     (type) == ICMP_IREQ || (type) == ICMP_IREQREPLY || 
     (type) == ICMP_MASKREQ || (type) == ICMP_MASKREPLY)
#define ICMP_ADVLEN(p)  
    (8 + ((p)->icmp_ip.ip_hl << 2) + 8)
#define ICMP_TSLEN      
    (8 + 3 * sizeof (n_time))
#define ICMP_ADVLENMIN  
    (8 + sizeof (struct ip) + 8)
#define ICMP_ECHOREPLY  0
#define ICMP_EXC_TTL    0
#define ICMP_NET_UNREACH  0
#define ICMP_REDIRECT_NET  0
#define ICMP_REDIRECT_NET  0
#define ICMP_TIMXCEED_INTRANS  0
#define ICMP_UNREACH_NET  0
#define ICMP_EXC_FRAGTIME  1
#define ICMP_HOST_UNREACH  1
#define ICMP_PARAMPROB_OPTABSENT  1
#define ICMP_REDIRECT_HOST  1
#define ICMP_TIMXCEED_REASS  1
#define ICMP_UNREACH_HOST  1
#define ICMP_HOST_ANO  10
#define ICMP_ROUTERSOLICIT  10
#define ICMP_UNREACH_HOST_PROHIB  10
#define ICMP_NET_UNR_TOS  11
#define ICMP_TIME_EXCEEDED  11
#define ICMP_TIMXCEED  11
#define ICMP_UNREACH_TOSNET  11
#define ICMP_HOST_UNR_TOS  12
#define ICMP_MASKLEN  12
#define ICMP_PARAMETERPROB  12
#define ICMP_PARAMPROB  12
#define ICMP_UNREACH_TOSHOST  12
#define ICMP_PKT_FILTERED  13
#define ICMP_TIMESTAMP  13
#define ICMP_TSTAMP  13
#define ICMP_UNREACH_FILTER_PROHIB  13
#define ICMP_PREC_VIOLATION  14
#define ICMP_TIMESTAMPREPLY  14
#define ICMP_TSTAMPREPLY  14
#define ICMP_UNREACH_HOST_PRECEDENCE  14
#define ICMP_INFO_REQUEST  15
#define ICMP_IREQ  15
#define ICMP_PREC_CUTOFF  15
#define ICMP_UNREACH_PRECEDENCE_CUTOFF  15
#define NR_ICMP_UNREACH  15
#define ICMP_INFO_REPLY  16
#define ICMP_IREQREPLY  16
#define ICMP_PREC_CUTOFF  15
#define ICMP_UNREACH_PRECEDENCE_CUTOFF  15
#define NR_ICMP_UNREACH  15
#define ICMP_INFO_REPLY  16
#define ICMP_IREQREPLY  16
```c
#define ICMP_ADDRESS 17
#define ICMP_MASKREQ 17
#define ICMP_ADDRESS_REPLY 18
#define ICMP_MASKREPLY 18
#define ICMP_MAXTYPE 18
#define NR_ICMP_TYPES 18
#define ICMP_UNREACH 2
#define ICMP_REDIRECT_TOSNET 2
#define ICMP_REDIR_NETTOS 2
#define ICMP_UNREACH_PROTOCOL 2
#define ICMP_DEST_UNREACH 3
#define ICMP_PORT_UNREACH 3
#define ICMP_REDIRECT_TOSHOST 3
#define ICMP_REDIR_HOSTTOS 3
#define ICMP_UNREACH 3
#define ICMP_UNREACH_PORT 3
#define ICMP_FRAG_NEEDED 4
#define ICMP_SOURCEQUENCH 4
#define ICMP_SOURCE_QUENCH 4
#define ICMP_UNREACH_NDFRAG 4
#define ICMP_REDIRECT 5
#define ICMP_SR_FAILED 5
#define ICMP_UNREACH_RAWFrag 5
#define ICMP_NET_UNKNOWN 6
#define ICMP_UNREACH_NET_UNKNOWN 6
#define ICMP_HOST_UNKNOWN 7
#define ICMP_UNREACH_HOST_UNKNOWN 7
#define ICMP_ECHO 8
#define ICMP_HOST_ISOLATED 8
#define ICMP_MINLEN 8
#define ICMP_UNREACH_ISOLATED 8
#define ICMP_NET_ANO 9
#define ICMP_ROUTERADVERT 9
#define ICMP_UNREACH_NET_PROHIB 9
#define icmp_data icmp_dun.id_data
#define icmp_ip icmp_dun.id_ip.idi_ip
#define icmp_mask icmp_dun.id_mask
#define icmp_radv icmp_dun.id_radv
#define icmp_otime icmp_dun.id_ts.its_otime
#define icmp_rtime icmp_dun.id_ts.its_rtime
#define icmp_ttime icmp_dun.id_ts.its_ttime
#define icmp_gwaddr icmp_hun.ih_gwaddr
#define icmp_id icmp_hun.ih_idseq.icd_id
#define icmp_seq icmp_hun.ih_idseq.icd_seq
#define icmp Nextmtu icmp_hun.ih_pmtu.ipm_nextmtu
#define icmp_pmvoid icmp_hun.ih_pmtu.ipm_void
#define icmp_pptr icmp_hun.ih_pptr
#define icmp_lifetime icmp_hun.ih_rtradv.irt_lifetime
#define icmp_num_addrs icmp_hun.ih_rtradv.irt_num_addrs
#define icmp_wpa icmp_hun.ih_rtradv.irt_wpa
#define icmp void icmp_hun.ih_void

struct icmphdr {
    u_int8_t type;
    u_int8_t code;
    u_int16_t checksum;
    union {
        struct {
            u_int16_t id;
            u_int16_t sequence;
        } echo;
        u_int32_t gateway;
        struct {
            u_int16_t __unused;
            u_int16_t mtu;
        } frag;
    }
};
```

© 2015 Linux Foundation
14.4.44 netinet/tcp.h

#define TCPOLEN_TSTAMP_APPA     (TCPOLEN_TIMESTAMP+2)
#define TCPOPT_TSTAMP_HDR       (TCPOPT_NOP<<24|TCPOPT_NOP<<16|
                            TCPOPT_TIMESTAMP<<8|TCPOLEN_TIMESTAMP)
#define TCPOPT_EOL      0
#define TCP_INFO        11
#define TCP_MAX_WINSHIFT        14
#define TCP_MAXSEG      2
#define TCP_MAXSEG    2
#define TCPOLEN_WINDOW  3
#define TCPOPT_WINDOW   3
#define TCP_CORK        3
#define TCPI_OPT_WSCALE 4
#define TCPOLEN_MAXSEG  4
#define TCPOPT_SACK_PERMITTED 4
#define TCP_KEEPIDLE    4
#define TCPKEEP_SACK    5
#define TCP_KEEPINTVL   5
#define TCP_MSS 512
#define SOL_TCP 6
#define TCP_KEEPCNT     6
#define TCP_MAXWIN 65535
#define TCP_SYNCNT 7
#define TCPI_OPT_ECN    8
#define TCPOPT_TIMESTAMP 8
#define TCP_LINGER2 8
#define TCP_DEFER_ACCEPT 9

enum tcp_ca_state {
    TCP_CA_Open,
    TCP_CA_Disorder,
    TCP_CA_CWR,
    TCP_CA_Recovery,
    TCP_CA_Loss
};

struct tcp_info {
    uint8_t tcpi_state;
    uint8_t tcpi_ca_state;
    uint8_t tcpi_retransmits;
    uint8_t tcpi_probes;
    uint8_t tcpi_backoff;
    uint8_t tcpi_options;
    uint8_t tcpi_snd_wscale:4;
    uint8_t tcpi_rcv_wscale:4;
    uint32_t tcpi_rto;
    uint32_t tcpi_ato;
    uint32_t tcpi_snd_mss;
    uint32_t tcpi_rcv_mss;
    uint32_t tcpi_unacked;
    uint32_t tcpi_sacked;
    uint32_t tcpi_lost;
    uint32_t tcpi_retrans;
    uint32_t tcpi_fackets;
    uint32_t tcpi_last_data_sent;
    uint32_t tcpi_last_ack_sent;
    uint32_t tcpi_last_data_recv;
    uint32_t tcpi_last_ack_recv;
    uint32_t tcpi_pmtu;
    uint32_t tcpi_rcv_ssthresh;
    uint32_t tcpi_snd_ssthresh;
    uint32_t tcpi_snd_cwnd;
    uint32_t tcpi_advmss;
    uint32_t tcpi_reordering;
};

enum {
    TCP_ESTABLISHED = 1,
    TCP_SYN_SENT = 2,
    TCP_SYN_RECV = 3,
    TCP_FIN_WAIT1 = 4,
    TCP_FIN_WAIT2 = 5,
    TCP_TIME_WAIT = 6,
    TCP_CLOSE = 7,
    TCP_CLOSE_WAIT = 8,

© 2015 Linux Foundation
TCP_LAST_ACK = 9,
TCP_LISTEN = 10,
TCP_CLOSING = 11

14.4.45 netinet/udp.h

#define SOL_UDP 17

struct udphdr {
    u_int16_t source;
    u_int16_t dest;
    u_int16_t len;
    u_int16_t check;
};

14.4.46 nl_types.h

#define NL_CAT_LOCALE   1
#define NL_SETD 1
typedef void *nl_catd;
typedef int nl_item;
extern int catclose(nl_catd __catalog);
extern char *catgets(nl_catd __catalog, int __set, int __number,
    const char *__string);
extern nl_catd catopen(const char *__cat_name, int __flag);

14.4.47 poll.h

extern int poll(struct pollfd *__fds, nfds_t __nfds, int __timeout);

14.4.48 pwd.h

struct passwd {
    char *pw_name;
    char *pw_passwd;
    uid_t pw_uid;
    gid_t pw_gid;
    char *pw_gecos;
    char *pw_dir;
    char *pw_shell;
};
extern void endpwent(void);
extern struct passwd *getpwent(void);
extern int getpwent_r(struct passwd *__resultbuf, char *__buffer,
    size_t __buflen, struct passwd **__result);
extern struct passwd *getpwnam(const char *__name);
extern int getpwnam_r(const char *__name, struct passwd *
    __resultbuf,
    char *__buffer, size_t __buflen,
    struct passwd **__result);
extern struct passwd *getpwuid(const char ___uid);
extern int getpwuid_r(uid_t __uid, struct passwd *__resultbuf,
    char *__buffer, size_t __buflen,
    struct passwd **__result);
extern void setpwent(void);
#define RE_DUP_MAX   (0x7fff)

typedef unsigned long int reg_syntax_t;

typedef struct re_pattern_buffer {
    unsigned char *buffer;
    unsigned long int allocated;
    unsigned long int used;
    reg_syntax_t syntax;
    char *fastmap;
    char *translate;
    size_t re_nsub;
    unsigned int can_be_null:1;
    unsigned int regs_allocated:2;
    unsigned int fastmap_accurate:1;
    unsigned int no_sub:1;
    unsigned int not_bol:1;
    unsigned int not_eol:1;
    unsigned int newline_anchor:1;
} regex_t;

typedef int regoff_t;

typedef struct {
    regoff_t rm_so;
    regoff_t rm_eo;
} regmatch_t;

#define REG_ICASE       (REG_EXTENDED<<1)
#define REG_NEWLINE     (REG_ICASE<<1)
#define REG_NOSUB       (REG_NEWLINE<<1)
#define REG_EXTENDED    1
#define REG_NOTEOL      (1<<1)
#define REG_NOTBOL      1

typedef enum {
    REG_ENOSYS = -1,
    REG_NOERROR = 0,
    REG_NOMATCH = 1,
    REG_BADPAT = 2,
    REG_ECOLLATE = 3,
    REG_ECTYPE = 4,
    REG_EESCAPE = 5,
    REG_ESUBREG = 6,
    REG_EBRACK = 7,
    REG_EPAREN = 8,
    REG_EBRACE = 9,
    REG_BADBR = 10,
    REG_ERANGE = 11,
    REG_ERANGE = 11,
    REG_ESPACE = 12,
    REG_BADPAT = 13,
    REG_EEND = 14,
    REG_ESIZE = 15,
    REG_ERPAREN = 16
} reg_errcode_t;

extern int regcomp(regex_t * __preg, const char *__pattern, int __cflags);
extern size_t regerror(int __errcode, const regex_t * __preg, char *__errbuf, size_t __errbuf_size);
extern int regexec(const regex_t * __preg, const char *__string, size_t __nmatch, regmatch_t __pmatch[], int __eflags);
extern void regfree(regex_t * __preg);
14.4.50 rpc/auth.h

#define auth_destroy(auth) (*((auth)->ah_ops->ah_destroy)(auth))

enum auth_stat {
    AUTH_OK = 0,
    AUTH_BADCRED = 1, /* bogus credentials (seal broken) */
    AUTH_REJECTEDCRED = 2, /* client should begin new session */
    AUTH_BADVERF = 3, /* bogus verifier (seal broken) */
    AUTH_REJECTEDVERF = 4, /* verifier expired or was replayed */
    AUTH_TOOWEAK = 5, /* Rpc calls return an enum clnt_stat */
    AUTH_INVALIDRESP = 6, /* bogus response verifier */
    AUTH_FAILED = 7 /* some unknown reason */
};

union des_block {
    struct {
        u_int32_t high;
        u_int32_t low;
    } key;
    char c[8];
};

struct opaque_auth {
    enum_t oa_flavor; /* flavor of auth */
    caddr_t oa_base; /* address of more auth stuff */
    u_int oa_length; /* not to exceed MAX_AUTH_BYTES */
};

typedef struct AUTH {
    struct opaque_auth ah_cred;
    struct opaque_auth ah_verf;
    union des_block ah_key;
    struct auth_ops *ah_ops;
    caddr_t ah_private;
} AUTH;

struct auth_ops {
    void (*ah_nextverf) (struct AUTH *);
    int (*ah_marshal) (struct AUTH *, XDR *); /* nextverf & serialize */
    int (*ah_validate) (struct AUTH *, struct opaque_auth *); /* validate verifier */
    int (*ah_refresh) (struct AUTH *); /* refresh credentials */
    void (*ah_destroy) (struct AUTH *); /* Rpc calls return an enum clnt_stat */
};

extern struct AUTH *authnone_create(void);
extern int key_decryptsession(char *, union des_block *);
extern bool_t xdr_opaque_auth(XDR *, struct opaque_auth *);

14.4.51 rpc/clnt.h

#define clnt_control(cl,rq,in) (*((cl)->cl_ops->cl_control)(cl,rq,in))
#define clnt_abort(rh) (*((rh)->cl_ops->cl_abort)(rh))
#define clnt_destroy(rh)        (*((rh)->cl_ops->cl_destroy)(rh))
#define  clnt_freeres(rh,xres,resp)       (*((rh)->cl_ops->cl_freeres)(rh,xres,resp))
#define clnt_geterr(rh,errp)    (*((rh)->cl_ops->cl_geterr)(rh, errp))

#define NULLPROC        ((u_long)0)     /* By convention, procedure 0 takes null arguments and returns */
#define CLSET_TIMEOUT   1       /* set timeout (timeval) */
#define CLGET_XID 10      /* Get xid */
#define CLSET_XID 11      /* Set xid */
#define CLSET_VERS 12     /* Get version number */
#define CLSET_VERS 13     /* Set version number */
#define CLGET_PROG 14     /* Get program number */
#define CLSET_PROG 15     /* Set program number */
#define CLGET_TIMEOUT 2   /* get timeout (timeval) */
#define CLGET_SERVER_ADDR 3 /* get server's address (sockaddr) */
#define CLSET_RETRY_TIMEOUT 4  /* set retry timeout (timeval) */
#define CLGET_RETRY_TIMEOUT 5 /* get retry timeout (timeval) */
#define CLGET_FD 6        /* get connections file descriptor */
#define CLGET_SVC_ADDR 7  /* get server's address (netbuf) */
#define CLSET_FD_CLOSE 8  /* close fd while clnt_destroy */
#define CLSET_FD_NCLOSE 9 /* Do not close fd while clnt_destroy */
#define clnt_call(rh, proc, xargs, argsp, xres, resp, secs)     
((*(rh)->cl_ops->cl_call)(rh, proc, xargs, argsp, xres, 
(resp, secs))

enum clnt_stat {
RPC_SUCCESS = 0,       /* call succeeded */
RPC_CANTENCODERARGS = 1,      /* can't encode arguments */
RPC_CANTDECODERES = 2,      /* can't decode results */
RPC_CANTSEND = 3,       /* failure in sending call */
RPC_CANTRECVR = 4,       /* failure in receiving result */
RPC_TIMEDOUT = 5,       /* call timed out */
RPC_VERSMISMATCH = 6,     /* rpc versions not compatible */
RPC_AUTHERROR = 7,       /* authentication error */
RPC_PROGRAMUNAVAIL = 8,   /* program not available */
RPC_PROGRAMVERSMismatch = 9,  /* program version mismatched */
RPC_PROCUNAVAIL = 10,     /* procedure unavailable */
RPC_CANTDECODEARGS = 11, /* decode arguments error */
RPC_SYSTEMERROR = 12,     /* generic "other problem" */
RPC_NOBROADCAST = 21,     /* Broadcasting not supported */
RPC_UNKNOWNHOST = 13,     /* unknown host name */
RPC_UNKNOWNPROTO = 17,    /* unknown protocol */
RPC_UNKNOWNADDR = 19,     /* Remote address unknown */
RPC_RPCBFailure = 14,     /* portmapper failed in its call */
RPC_PROGNOTREGISTERED = 15, /* remote program is not registered */
RPC_N2AXLATEFAILURE = 22, /* Name to addr translation failed */
RPC_FAILED = 16,
RPC_INTR = 18,
RPC_TLIERROR = 20,
RPC_UDERROR = 23,
RPC_INPROGRESS = 24,
RPC_STALERACHANDLE = 25
};
struct rpc_err {  
   enum clnt_stat re_status;
   union {


© 2015 Linux Foundation
```c
typedef struct CLIENT {
    struct AUTH *cl_auth;
    struct clnt_ops *cl_ops;
    caddr_t cl_private;
} CLIENT;

struct clnt_ops {
    enum clnt_stat (*cl_call) (struct CLIENT *, u_long, xdrproc_t, caddr_t, xdrproc_t, caddr_t, struct timeval);
    void (*cl_abort) (void);
    void (*cl_geterr) (struct CLIENT *, xdrproc_t, caddr_t);
    bool_t(*cl_freeses) (struct CLIENT *, xdrproc_t, caddr_t);
    void (*cl_destroy) (struct CLIENT *, int, char *);
};

extern int callrpc(const char *__host, const u_long __prognum, const u_long __versnum, const u_long __procnum, const xdrproc_t __inproc, const char *__in, const xdrproc_t __outproc, char *__out);

extern struct CLIENT *clnt_create(const char *__host, const u_long __prog, const u_long __vers, const char *__prot);
extern void clnt_pcreateerror(const char *__msg);
extern void clnt_perrno(enum clnt_stat __num);
extern void clnt_perror(struct CLIENT *__clnt, const char *__msg);
extern char *clnt_spcreateerror(const char *__msg);
extern char *clnt_sperrno(enum clnt_stat __num);
extern char *clnt_sperror(struct CLIENT *__clnt, const char *__msg);
extern struct CLIENT *clnttcp_create(struct sockaddr_in *__raddr, u_long __prog, u_long __version, struct timeval __wait_resend, int *__sockp, u_int __sendsz, u_int __recvsz);
extern struct CLIENT *clntudp_bufcreate(struct sockaddr_in *__raddr, u_long __program, u_long __version, struct timeval __wait_resend, int *__sockp, u_int __sendsz, u_int __recvsz);
extern struct CLIENT *clntudp_create(struct sockaddr_in *__raddr, u_long __program, u_long __version, struct timeval __wait_resend, int *__sockp, u_int __sendsz, u_int __recvsz);
```
\_version, \_wait_resend, \_

14.4.52 rpc/pmap_clnt.h

extern u\_short pmap\_getport(struct sockaddr\_in *\_address, \nconst u\_long \_program, const u\_long \_version, \n\_protocol);\n
extern bool\_t pmap\_set(const u\_long \_program, const u\_long \_vers, \nint \_protocol, u\_short \_port);\n
extern bool\_t pmap\_unset(u\_long \_program, u\_long \_vers);

14.4.53 rpc/rpc\_msg.h

enum msg\_type {\n  CALL = 0,\n  REPLY = 1\n};\n
enum reply\_stat {\n  MSG\_ACCEPTED = 0,\n  MSG\_DENIED = 1\n};\n
enum accept\_stat {\n  SUCCESS = 0,\n  PROG\_UNAVAIL = 1,\n  PROG\_MISMATCH = 2,\n  PROC\_UNAVAIL = 3,\n  GARBAGE\_ARGS = 4,\n  SYSTEM\_ERR = 5\n};\n
enum reject\_stat {\n  RPC\_MISMATCH = 0,\n  AUTH\_ERROR = 1\n};\n
#define ar\_results ru.AR\_results\n#define ar\_vers ru.AR\_versions

struct accepted\_reply {\n  struct opaque\_auth ar\_verf;\n  enum accept\_stat ar\_stat;\n  union {\n    struct {\n      unsigned long int low;\n      unsigned long int high;\n    } AR\_versions;\n    struct {\n      caddr\_t where;\n      xdrproc\_t proc;\n    } AR\_results;\n  } ru;\n};\n
#define rj\_vers ru.RJ\_versions\n#define rj\_why ru.RJ\_why

struct rejected\_reply {\n  enum reject\_stat rj\_stat;\n  union {\n
struct {
  unsigned long int low;
  unsigned long int high;
  RJ_versions;
  enum auth_stat RJ_why; /* why authentication did not work */
} ru;

#define rp_acpt ru.RP_ar
#define rp_rjct ru.RP_dr

struct reply_body {
  enum reply_stat rp_stat;
  union {
    struct accepted_reply RP_ar;
    struct rejected_reply RP_dr;
  } ru;
};

#define rm_call ru.RM_cmb
#define rm_reply ru.RM_rmb
#define accepted_reply ru.RM_rmb.ru.RP_ar
#define rjected_reply ru.RM_rmb.ru.RP_dr

struct call_body {
  unsigned long int cb_rpcvers; /* must be equal to two */
  unsigned long int cb_prog;
  unsigned long int cb_ver;
  unsigned long int cb_proc;
  struct opaque_auth cb_cred;
  struct opaque_auth cb_verf; /* protocol specific - provided by client */
};

#define rm_call ru.RM_cmb
#define rm_reply ru.RM_rmb
#define accepted_reply ru.RM_rmb.ru.RP_ar
#define rejected_reply ru.RM_rmb.ru.RP_dr

struct rpc_msg {
  unsigned long int rm_xid;
  enum msg_type rm_direction;
  union {
    struct call_body RM_cmb;
    struct reply_body RM_rmb;
  } ru;
};

extern bool_t xdr_accepted_reply(XDR *, struct accepted_reply *);
extern bool_t xdr_callhdr(XDR * __xdrs, struct rpc_msg *__cmsg);
extern bool_t xdr_callmsg(XDR * __xdrs, struct rpc_msg *__cmsg);
extern bool_t xdr_rejected_reply(XDR *, struct rejected_reply *);
extern bool_t xdr_replymsg(XDR * __xdrs, struct rpc_msg *__rmsg);

14.4.54 rpc/svc.h

#define svc_getcaller(x)        (&(x)->xp_raddr)
#define svc_destroy(xprt)        (*(xprt)->xp_ops->xp_destroy)(xprt)
#define svc_recv(xprt,msg)       (*(xprt)->xp_ops->xp_recv)(xprt, (msg))
#define svc_reply(xprt,msg)      (*(xprt)->xp_ops->xp_reply)(xprt, (msg))
#define svc_stat(xprt)  (*(xprt)->xp_ops->xp_stat)(xprt)
#define RPC_ANYSOCK     -1
#define svc_freeargs(xprt,xargs, argsp) \
  (*(xprt)->xp_ops->xp_freeargs)((xprt), (xargs), (argsp))
#define svc_getargs(xprt,xargs, argsp)  \
  (*(xprt)->xp_ops->xp_getargs)((xprt), (xargs), (argsp))

128
enum xprt_stat {
    XPRT_DIED,
    XPRT_MOREREQS,
    XPRT_IDLE
};

typedef struct SVCXPRT {
    int xp_sock;
    u_short xp_port;
    struct xp_ops *xp_ops;
    int xp_addrlen;
    struct sockaddr_in xp_raddr;
    struct opaque_auth xp_verf;
    caddr_t xp_p1;
    caddr_t xp_p2;
    char xp_pad[256];
} SVCXPRT;

struct svc_req {
    rpcprog_t rq_prog;
    rpcvers_t rq_vers;
    rpcproc_t rq_proc;
    struct opaque_auth rq_cred;
    caddr_t rq_clntcred;
    SVCXPRT *rq_xprt;
};

typedef void (*__dispatch_fn_t) (struct svc_req *, SVCXPRT *);

struct xp_ops {
    bool_t(*xp_recv) (SVCXPRT * __xprt, struct rpc_msg * __msg);
    enum xprt_stat (*xp_stat) (SVCXPRT * __xprt);
    bool_t(*xp_getargs) (SVCXPRT * __xprt, xdrproc_t __xdr_args,
        caddr_t args_ptr);
    bool_t(*xp_reply) (SVCXPRT * __xprt, struct rpc_msg * __msg);
    bool_t(*xp_freeargs) (SVCXPRT * __xprt, xdrproc_t __xdr_args,
        caddr_t args_ptr);
    void (*xp_destroy) (SVCXPRT * __xprt);
};

extern void svc_getreqset(fd_set * __readfds);
extern bool_t svc_register(SVCXPRT * __xprt, rpcprog_t __prog,
    __dispatch_fn_t __dispatch,
    rpcprot_t __protocol);
extern void svc_run(void);
extern bool_t svc_sendreply(SVCXPRT * xprt, xdrproc_t __xdr_results,
    caddr_t __xdr_location);
extern void svcerr_auth(SVCXPRT * __xprt, enum auth_stat __why);
extern void svcerr_decode(SVCXPRT * __xprt);
extern void svcerr_noprocp(SVCXPRT * __xprt);
extern void svcerr_noprog(SVCXPRT * __xprt);
extern void svcerrProgvers(SVCXPRT * __xprt, rpcvers_t __low_vers,
    rpcvers_t __high_vers);
extern void svcerr_systemerr(SVCXPRT * __xprt);
extern void svcerr_weakauth(SVCXPRT * __xprt);
extern SVCXPRT *svcfd_create(int, unsigned int, unsigned int);
extern SVCXPRT *svcraw_create(void);
extern SVCXPRT *svctcp_create(int __sock, u_int __sendsize,
    u_int __recvsize);
extern SVCXPRT *svcsudp_create(int __sock);
14.4.55 `rpc/types.h`

typedef int bool_t;
typedef int enum_t;
typedef unsigned long int rpcprog_t;
typedef unsigned long int rpcvers_t;
typedef unsigned long int rpcproc_t;
typedef unsigned long int rpcprot_t;

14.4.56 `rpc/xdr.h`

#define XDR_DESTROY(xdrs)   \
   do { if ((xdrs)->x_ops->x_destroy) (*(xdrs)->x_ops->x_destroy) (xdrs); } while (0)
#define xdr_destroy(xdrs)   \
   do { if ((xdrs)->x_ops->x_destroy) (*(xdrs)->x_ops->x_destroy) (xdrs); } while (0)
#define XDR_GETBYTES(xdrs,addr,len)      (*(xdrs)->x_ops->x_getbytes)(xdrs, addr, len)
#define xdr_getbytes(xdrs,addr,len)      (*(xdrs)->x_ops->x_getbytes)(xdrs, addr, len)
#define XDR_GETINT32(xdrs,int32p)         (*(xdrs)->x_ops->x_getint32)(xdrs, int32p)
#define xdr_getint32(xdrs,int32p)         (*(xdrs)->x_ops->x_getint32)(xdrs, int32p)
#define XDR_GETLONG(xdrs,longp)           (*(xdrs)->x_ops->x_getlong)(xdrs, longp)
#define xdr_getlong(xdrs,longp)           (*(xdrs)->x_ops->x_getlong)(xdrs, longp)
#define XDR_GETPOS(xdrs)                  (*(xdrs)->x_ops->x_getposn)(xdrs)
#define xdr_getpos(xdrs)                  (*(xdrs)->x_ops->x_getposn)(xdrs)
#define XDR_INLINE(xdrs,len)              (*(xdrs)->x_ops->x_inline)(xdrs, len)
#define xdr_inline(xdrs,len)              (*(xdrs)->x_ops->x_inline)(xdrs, len)
#define XDR_PUTBYTES(xdrs,addr,len)       (*(xdrs)->x_ops->x_putbytes)(xdrs, addr, len)
#define xdr_putbytes(xdrs,addr,len)       (*(xdrs)->x_ops->x_putbytes)(xdrs, addr, len)
#define XDR_PUTINT32(xdrs,int32p)         (*(xdrs)->x_ops->x_putint32)(xdrs, int32p)
#define xdr_putint32(xdrs,int32p)         (*(xdrs)->x_ops->x_putint32)(xdrs, int32p)
#define XDR_PUTLONG(xdrs,longp)           (*(xdrs)->x_ops->x_putlong)(xdrs, longp)
#define xdr_putlong(xdrs,longp)           (*(xdrs)->x_ops->x_putlong)(xdrs, longp)
#define XDR_SETPOS(xdrs,pos)              (*(xdrs)->x_ops->x_setposn)(xdrs, pos)
#define xdr_setpos(xdrs,pos)              (*(xdrs)->x_ops->x_setposn)(xdrs, pos)

enum xdr_op {   
   XDR_ENCODE,  
   XDR_DECODE,  
   XDR_FREE  
};
typedef struct XDR {   
   enum xdr_op x_op;
struct xdr_ops *x_ops;
caddr_t x_public;
caddr_t x_private;
caddr_t x_base;
int x_handy;
} XDR;

struct xdr_ops {
    bool_t(*x_getlong) (XDR * __xdrs, long int *__lp);
    bool_t(*x_putlong) (XDR * __xdrs, long int *__lp);
    bool_t(*x_getbytes) (XDR * __xdrs, caddr_t __addr, u_int __len);
    bool_t(*x_putbytes) (XDR * __xdrs, char *__addr, u_int __len);
    u_int(*x_getpostn) (XDR * __xdrs);
    bool_t(*x_setpostn) (XDR * __xdrs, u_int __pos);
    int32_t(*(x_inline) (XDR * __xdrs, int __len);
    void (*x_destroy) (XDR * __xdrs);
    bool_t(*x_getint32) (XDR * __xdrs, int32_t * __ip);
    bool_t(*x_putint32) (XDR * __xdrs, int32_t * __ip);
};

typedef bool_t(*xdrproc_t) (XDR *, void *, ...);

struct xdr_discrim {
    int value;
    xdrproc_t proc;
};

extern bool_t xdr_array(XDR * _xdrs, caddr_t * __addrp, u_int * __sizep,
                        u_int __maxsize, u_int __elsize,
                        xdrproc_t __elproc);

extern bool_t xdr_bool(XDR * __xdrs, bool_t * __bp);

extern bool_t xdr_bytes(XDR * __xdrs, char **__cpp, u_int * __sizep,
                        u_int __maxsize);

extern bool_t xdr_char(XDR * __xdrs, char *__cp);

extern bool_t xdr_double(XDR * __xdrs, double *__dp);

extern bool_t xdr_enum(XDR * __xdrs, enum_t * __ep);

extern bool_t xdr_float(XDR * __xdrs, float *__fp);

extern void xdr_free(xdrproc_t __proc, char *__objp);

extern bool_t xdr_int(XDR * __xdrs, int *__ip);

extern bool_t xdr_long(XDR * __xdrs, long int *__lp);

extern bool_t xdr_opaque(XDR * __xdrs, caddr_t __cp, u_int __cnt);

extern bool_t xdr_pointer(XDR * __xdrs, char **__objpp, u_int __obj_size,
                           xdrproc_t __xdr_obj);

extern bool_t xdr_reference(XDR * __xdrs, caddr_t __xpp, u_int __size,
                             xdrproc_t __proc);

extern bool_t xdr_short(XDR * __xdrs, short *__sp);

extern bool_t xdr_string(XDR * __xdrs, char **__cpp, u_int __maxsize);

extern bool_t xdr_u_char(XDR * __xdrs, u_char *__cp);

extern bool_t xdr_u_int(XDR * __xdrs, u_int *__up);

extern bool_t xdr_u_long(XDR * __xdrs, u_long *__ulp);

extern bool_t xdr_u_short(XDR * __xdrs, u_short *__usp);

extern bool_t xdr_union(XDR * __xdrs, enum_t *__dscmp, char *__unp,
                        const struct xdr_discrim *__choices,
                        xdrproc_t __default);

extern bool_t xdr_vector(XDR * __xdrs, char *basep, u_int __nelem,
                          u_int __elem_size, xdrproc_t __xdr_elem);

extern bool_t xdr_void(void);
extern bool_t xdr_wrapstring(XDR *__xdrs, char **__cpp);
extern void xdrmem_create(XDR *__xdrs, caddr_t __addr, u_int __size,
                        enum xdr_op __xop);
extern void xdrrec_create(XDR *__xdrs, u_int __sendsize, u_int __recvsize,
                        caddr_t __tcp_handle, int (*__readit)(char *,
                        char *,
                        int),
                        int (*__writeit)(char *, char *,
                        int));
extern bool_t xdrrec_endofrecord(XDR *__xdrs, bool_t __sendnow);
extern bool_t xdrrec_eof(XDR *__xdrs);
extern bool_t xdrrec_skiprecord(XDR *__xdrs);
extern void xdrstdio_create(XDR *__xdrs, FILE *__file,
                        enum xdr_op __xop);

14.4.57 sched.h

#define __CPU_ALLOC_SIZE(count) (((((count) + __NCPUBITS - 1) /
                              __NCPUBITS) * 8)
#define __CPUELT(cpu) ((cpu) / __NCPUBITS)
#define __CPUMASK(cpu) ((__cpu_mask) 1 << ((cpu) % __NCPUBITS))
#define __NCPUBITS (8 * sizeof (__cpu_mask))
#define SCHED_OTHER 0
#define SCHED_FIFO 1
#define __CPU_SETSIZE 1024
#define SCHED_RR 2
#define _extension_
#define __CPU_OP_S(setsize, destset, srcset1, srcset2, op)      
       (__extension__
       {
       cpu_set_t *__dest = (destset);
       cpu_set_t *__arr1 = (srcset1);
       cpu_set_t *__arr2 = (srcset2);
       size_t __imax = (setsize) / sizeof (__cpu_mask);
       size_t __i;
       for (__i = 0; __i < __imax; ++__i)
       __dest->__bits[__i] = __arr1->__bits[__i] op __arr2->
       >__bits[__i];
       __dest; })
#define __CPU_SET_S(cpu, setsize, cpusetp)      
       (__extension__
       {
       size_t __cpu = (cpu);
       __cpu < 8 * (setsize) 
       ? ((cpusetp)->__bits[__CPUELT ((__cpu)] |= __CPUMASK ((__cpu]) : 0;
       })
#define __CPU_ISSET_S(cpu, setsize, cpusetp)    
       (__extension__
       {
       size_t __cpu = (cpu);
       __cpu < 8 * (setsize)
       ? ((cpusetp)->__bits[__CPUELT ((__cpu)] & __CPUMASK
       ((__cpu]) != 0 
       : 0;
       })
#define __CPU_CLR_S(cpu, setsize, cpusetp)      
       (__extension__
       {
       size_t __cpu = (cpu);
       __cpu < 8 * (setsize)
       ? ((cpusetp)->__bits[__CPUELT ((__cpu)] & __CPUMASK
       ((__cpu]) : 0;
       })
#define __CPU_ZERO_S(setsize, cpusetp)  
do {
       size_t __i;
       size_t __imax = (setsize) / sizeof (__cpu_mask);
       \
#define CPU_ALLOC_SIZE(count)   __CPU_ALLOC_SIZE (count)
#define  CPU_CLR(cpu,  cpusetp)    __CPU_CLR_S  (cpu,  sizeof (cpu_set_t), cpusetp)
#define  CPU_ISSET(cpu,  cpusetp)  __CPU_ISSET_S  (cpu,  sizeof (cpu_set_t), cpusetp)
#define  CPU_AND_S(setsize,  destset,  srcset1,  srcset2)
  __CPU_OP_S (setsize, destset, srcset1, srcset2, &)
#define  CPU_XOR_S(setsize,  destset,  srcset1,  srcset2)
  __CPU_OP_S (setsize, destset, srcset1, srcset2, ^)
#define  CPU_OR_S(setsize,  destset,  srcset1,  srcset2)
  __CPU_OP_S (setsize, destset, srcset1, srcset2, |)
#define CPU_AND(destset, srcset1, srcset2)      __CPU_OP_S (sizeof (cpu_set_t), destset, srcset1, srcset2, &)
#define CPU_XOR(destset, srcset1, srcset2)      __CPU_OP_S (sizeof (cpu_set_t), destset, srcset1, srcset2, ^)
#define CPU_OR(destset, srcset1, srcset2)       __CPU_OP_S (sizeof (cpu_set_t), destset, srcset1, srcset2, |)
#define CPU_SETSIZE     __CPU_SETSIZE
#define  CPU_SET(cpu,  cpusetp)    __CPU_SET_S  (cpu,  sizeof (cpu_set_t), cpusetp)
#define CPU_ZERO(cpusetp)       __CPU_ZERO_S (sizeof (cpu_set_t), cpusetp)

struct sched_param {
  int sched_priority;
};
typedef unsigned long int __cpu_mask;
typedef struct {
  __cpu_mask __bits[__CPU_SETSIZE / __NCPUBITS];
  } cpu_set_t;
extern int sched_get_priority_max(int __algorithm);
extern int sched_get_priority_min(int __algorithm);
extern int sched_getaffinity(pid_t __pid, size_t __cpusetsize,
                               cpu_set_t * __cpuset);
extern int sched_getparam(pid_t __pid, struct sched_param *
                            __param);
extern int sched_setscheduler(pid_t __pid);
extern int sched_rr_get_interval(pid_t __pid, struct timespec *
                                t);
extern int sched_setaffinity(pid_t __pid, size_t __cpusetsize,
                             const cpu_set_t * __cpuset);
extern int sched_setparam(pid_t __pid, const struct sched_param *
                          __param);
extern int sched_setscheduler(pid_t __pid, int __policy,
                              const struct sched_param * __param);
extern int sched_yield(void);

14.4.58 search.h

typedef struct entry {
  char *key;
  void *data;
} ENTRY;
typedef enum {
  FIND,
  ENTER
} ACTION;
struct _ENTRY;
typedef enum {
  preorder,
  postorder,
endorder,
leaf
} VISIT;
struct hsearch_data {
    struct _ENTRY *table;
    unsigned int size;
    unsigned int filled;
};

typedef void (*__action_fn_t) (const void *__nodep, VISIT __value,
    int __level);
extern int hcreate(size_t __nel);
extern int hcreate_r(size_t __nel, struct hsearch_data *__htab);
extern void hdestroy(void);
extern void hdestroy_r(struct hsearch_data *__htab);
extern ENTRY *hsearch(ENTRY __item, ACTION __action);
extern int hsearch_r(ENTRY __item, ACTION __action, ENTRY * __retval,
    struct hsearch_data *__htab);
extern void insque(void *__elem, void *__prev);
extern void *lfind(const void *__key, const void *__base, size_t *__nmemb,
    size_t __size, __compar_fn_t __compar);
extern void *lsearch(const void *__key, void *__base, size_t *__nmemb,
    size_t __size, __compar_fn_t __compar);
extern void remque(void *__elem);
extern void *tdelete(const void *__key, void **__rootp,
    __compar_fn_t __compar);
extern void *tfind(const void *__key, void *const *__rootp,
    __compar_fn_t __compar);
extern void *tsearch(const void *__key, void *const *__rootp,
    __compar_fn_t __compar);
extern void twalk(const void *__root, __action_fn_t __action);

14.4.59 setjmp.h
#define setjmp(env)     _setjmp(env)
#define sigsetjmp(a,b)  __sigsetjmp(a,b)
struct __jmp_buf_tag {
    __jmp_buf __jmpbuf;
    int __mask_was_saved;
    sigset_t __saved_mask;
};
typedef struct __jmp_buf_tag jmp_buf[1];
typedef jmp_buf sigjmp_buf;
extern int __sigsetjmp(jmp_buf __env, int __savemask);
extern void _longjmp(jmp_buf __env, int __val);
extern void siglongjmp(sigjmp_buf __env, int __val);

14.4.60 signal.h
#define sigpause __xpg_sigpause
#define _SIGSET_NWORDS  (1024/(8*sizeof(unsigned long)))
#define SIGRTMAX        (__libc_current_sigrtmax ()
#define SIGRTMIN        (__libc_current_sigrtmin ()
#define NSIG    65
#define SIG_BLOCK 0    /* Block signals. */
#define SIG_UNBLOCK 1   /* Unblock signals. */
#define SIG_SETMASK 2   /* Set the set of blocked
signals. */

typedef int sig_atomic_t;

typedef void (*sighandler_t) (int);

#define SIG_HOLD ((sighandler_t) 2)    /* Request that
signal be held. */
#define SIG_DFL ((sighandler_t)0)     /* Request for default
signal handling. */
#define SIG_IGN ((sighandler_t)1)     /* Request that signal be
ignored. */
#define SIG_ERR ((sighandler_t)-1)    /* Return value from
signal() in case of error. */

#define SIGHUP  1               /* Hangup. */
#define SIGINT  2               /* Terminal interrupt signal. */
#define SIGQUIT 3               /* Terminal quit signal. */
#define SIGILL  4               /* Illegal instruction. */
#define SIGTRAP 5               /* Trace/breakpoint trap. */
#define SIGABRT 6               /* Process abort signal. */
#define SIGIOT  6               /* IOT trap */
#define SIGBUS  7               /* Access to an undefined portion
of a memory object. */
#define SIGFPE  8               /* Erroneous arithmetic
operation. */
#define SIGILL  9               /* Kill (cannot be caught or
ignored). */
#define SIGUSR1 10              /* User-defined signal 1. */
#define SIGSEGV 11              /* Invalid memory reference. */
#define SIGUSR2 12              /* User-defined signal 2. */
#define SIGPIPE 13              /* Write on a pipe with no one
to read it. */
#define SIGALRM 14              /* Alarm clock. */
#define SIGTERM 15              /* Termination signal. */
#define SIGSTKFLT 16            /* Stack fault. */
#define SIGCHLD 17              /* Child process terminated,
stopped, or continued. */
#define SIGCONT 18              /* Continue executing, if
stopped. */
#define SIGSTOP 19              /* Stop executing (cannot be
cought or ignored). */
#define SIGTSTP 20              /* Terminal stop signal. */
#define SIGTTIN 21              /* Background process attempting
read. */
#define SIGTTOU 22              /* Background process attempting
write. */
#define SIGURG  23              /* High bandwidth data is
available at a socket. */
#define SIGXCPU 24              /* CPU time limit exceeded. */
#define SIGXFSZ 25              /* File size limit exceeded. */
#define SIGVTALRM 26            /* Virtual timer expired. */
#define SIGPROF 27              /* Profiling timer expired. */
#define SIGWINCH 28             /* Window size change. */
#define SIGIO  29               /* I/O now possible. */
#define SIGPOLL SIGIO           /* Pollable event. */
#define SIGPWR  30              /* Power failure restart */
#define SIGSYS  31              /* Bad system call. */
#define SIGUNUSED 31

#define SV_ONSTACK (1<<0)    /* Take the signal on the signal
stack. */

© 2015 Linux Foundation
#define SV_INTERRUPT (1<<1) /* Do not restart system calls. */
#define SV_RESETHAND (1<<2) /* Reset handler to SIG_DFL on receipt. */

typedef union sigval {
    int sival_int;
    void *sival_ptr;
} sigval_t;

#define SIGEV_SIGNAL 0       /* Notify via signal. */
#define SIGEV_NONE 1         /* Other notification: meaningless. */
#define SIGEV_THREAD 2       /* Deliver via thread creation. */
#define SIGEV_MAX_SIZE 64

typedef struct sigevent {
    sigval_t sigev_value;
    int sigev_signo;
    int sigev_notify;
    union {
        int _pad[SIGEV_PAD_SIZE];
        struct {
            void (*_function) (sigval_t);
            void * _attribute;
        } _sigev_thread;
    } _sigev_un;
} sigevent_t;

#define SI_MAX_SIZE 128
#define si_pid _sifields._kill._pid
#define si_uid _sifields._kill._uid
#define si_value _sifields._rt._sigval
#define si_int _sifields._rt._sigval.sival_int
#define si_ptr _sifields._rt._sigval.sival_ptr
#define si_status _sifields._sigchld._status
#define si_stime _sifields._sigchld._stime
#define si_utime _sifields._sigchld._utime
#define si_addr _sifields._sigfault._addr
#define si_band _sifields._sigpoll._band
#define si_fd _sifields._sigpoll._fd
#define si_timer1 _sifields._timer._timer1
#define si_timer2 _sifields._timer._timer2
#define sigev_notify_attributes _sigev_un._sigev_thread._attribute
#define sigev_notify_function _sigev_un._sigev_thread._function

typedef struct siginfo {
    int si_signo;               /* Signal number. */
    int si_errno;
    int si_code;                /* Signal code. */
    union {
        int _pad[SI_PAD_SIZE];
        struct {
            pid_t _pid;
            uid_t _uid;
        } _kill;
        struct {
            unsigned int _timer1;
            unsigned int _timer2;
        } _timer;
    } _sifields;
} siginfo_t;


```c
} __rt;
struct {
    pid_t __pid;
    uid_t __uid;
    int __status;
    clock_t __utime;
    clock_t __stime;
} __sigchld;
struct {
    void *__addr;
} __sigfault;
struct {
    int __band;
    int __fd;
} __sigpoll;
} __sifields;
} siginfo_t;

#define SI_QUEUE -1      /* Sent by sigqueue. */
#define SI_TIMER -2      /* Sent by timer expiration. */
#define SI_MESGQ -3      /* Sent by real time mesq state change. */
#define SI_ASYNCIO -4     /* Sent by AIO completion. */
#define SI_SIGIO -5      /* Sent by queued SIGIO. */
#define SI_TKILL -6      /* Sent by tkill. */
#define SI_ASYNCNL -60    /* Sent by async name lookup completion. */
#define SI_USER 0         /* Sent by kill, sigsend, raise. */
#define SI_KERNEL 0x80    /* Sent by kernel. */

#define ILL_ILLOPC 1       /* Illegal opcode. */
#define ILL_ILLOPN 2       /* Illegal operand. */
#define ILL_ILLADR 3       /* Illegal addressing mode. */
#define ILL_ILLTRP 4       /* Illegal trap. */
#define ILL_PRVOPC 5       /* Privileged opcode. */
#define ILL_PRVREG 6       /* Privileged register. */
#define ILL_COPROC 7       /* Coprocessor error. */
#define ILL_BADSTK 8       /* Internal stack error. */
#define FPE_INTDIV 1       /* Integer divide by zero. */
#define FPE_INTOVF 2       /* Integer overflow. */
#define FPE_FLTDIV 3       /* Floating-point divide by zero. */
#define FPE_FLTOVF 4       /* Floating-point overflow. */
#define FPE_FLTUND 5       /* Floating-point underflow. */
#define FPE_FLTRES 6       /* Floating-point inexact result. */
#define FPE_FLTINV 7       /* Invalid floating-point operation. */
#define FPE_FLTSUB 8       /* Subscript out of range. */
#define SEGV_MAPERR 1      /* Address not mapped to object. */
#define SEGV_ACCERR 2      /* Invalid permissions for mapped object. */

#define BUS_ADRALN 1       /* Invalid address alignment. */
#define BUS_ADRERR 2       /* Nonexistent physical address. */
#define BUS_OBJERR 3       /* Object-specific hardware error. */
#define TRAP_BRKPT 1       /* Process breakpoint. */
#define TRAP_TRACE 2       /* Process trace trap. */
```
#define CLD_EXITED      1       /* Child has exited. */
#define CLD_KILLED      2       /* Child has terminated 
    abnormally and did not create a core file. */
#define CLD_DUMPED      3       /* Child has terminated 
    abnormally and created a core file. */
#define CLD_TRAPPED     4       /* Traced child has trapped. */
#define CLD_STOPPED     5       /* Child has stopped. */
#define CLD_CONTINUED   6       /* Stopped child has continued. */

#define POLL_IN 1               /* Data input available. */
#define POLL_OUT        2       /* Output buffers available. */
#define POLL_MSG        3       /* Input message available. */
#define POLL_ERR        4       /* I/O error. */
#define POLL_PRI        5       /* High priority input available. */
#define POLL_HUP        6       /* Device disconnected. */

typedef struct {
    unsigned long int sig[eshire_wordsize];
} sigset_t;

#define SA_INTERRUPT    0x20000000
#define sa_handler      __sigaction_handler._sa_handler
#define sa_sigaction    __sigaction_handler._sa_sigaction
#define SA_ONSTACK      0x08000000      /* Use signal stack by
    using `sa_restorer`. */
#define SA_RESETHAND    0x80000000      /* Reset to SIG_DFL on
    entry to handler. */
#define SA_NOCLDSTOP    0x00000001      /* Don't send SIGCHLD
    when children stop. */
#define SA_SIGINFO      0x00000004      /* Invoke signal-catching
    function with three arguments instead of one. */
#define SA_NODEFER      0x40000000      /* Don't automatically
    block the signal when its handler is being executed. */
#define SA_RESTART      0x10000000      /* Restart syscall on
    signal return. */
#define SA_NOCLDWAIT    0x00000002      /* Don't create zombie on
    child death. */
#define SA_NOMASK       SA_NODEFER
#define SA_ONESHOT      SA_RESETHAND
typedef struct sigaltstack {
    void *ss_sp;
    int ss_flags;
    size_t ss_size;
} stack_t;

#define SS_ONSTACK      1
#define SS_DISABLE      2

extern int __libc_current_sigrtmax(void);
extern int __libc_current_sigrtmin(void);
extern sighandler_t __sysv_signal(int __sig, sighandler_t __handler);
extern int __xpg_sigpause(int);
extern char *const _sys_siglist[];
extern sighandler_t bsd_signal(int __sig, sighandler_t __handler);
extern int kill(pid_t __pid, int __sig);
extern int killpg(pid_t __pgrp, int __sig);
extern void psiginfo(const siginfo_t * pinfo, const char *message);
extern void psignal(int __sig, const char *__s);
extern int pthread_kill(pthread_t, int);
extern int pthread_sigmask(int, const sigset_t *, sigset_t *);
extern int raise(int __sig);
extern int sigaction(int __sig, const struct sigaction *__act,
                     struct sigaction *__oact);
extern int sigaddset(sigset_t *__set, int __signo);
extern int sigaltstack(const struct sigaltstack *__ss,
                        struct sigaltstack *__oss);
extern int sigandset(sigset_t *__set, const sigset_t *__left,
                     const sigset_t *__right);
extern int sigdelset(sigset_t *__set, int __signo);
extern int sigemptyset(sigset_t *__set);
extern int sigfillset(sigset_t *__set);
extern int sighold(int __sig);
extern int sigignore(int __sig);
extern int siginterrupt(int __sig, int __interrupt);
extern int sigisemptyset(const sigset_t *__set);
extern int sigismember(const sigset_t *__set, int __signo);
extern sighandler_t signal(int __sig, sighandler_t __handler);
extern int sigorset(sigset_t *__set, const sigset_t *__left,
                    const sigset_t *__right);
extern int sigpending(sigset_t *__set);
extern int sigprocmask(int __how, const sigset_t *__set,
                       sigset_t *__oset);
extern int sigqueue(pid_t __pid, int __sig, const union sigval __val);
extern int sigrelse(int __sig);
extern int sigreturn(struct sigcontext *__scp);
extern sighandler_t sigset(int __sig, sighandler_t __handler);
extern int sigsuspend(const sigset_t *__set);
extern int sigtimedwait(const sigset_t *__set, siginfo_t *__info,
                         const struct timespec *__timeout);
extern int sigwait(const sigset_t *__set, int *__sig);
extern int sigwaitinfo(const sigset_t *__set, siginfo_t *__info);

14.4.61 spawn.h

#define POSIX_SPAWN_RESETIDS    0x01
#define POSIX_SPAWN_SETGROUPE 0x02
#define POSIX_SPAWN_SETSIGDEF  0x04
#define POSIX_SPAWN_SETSIGMASK 0x08
#define POSIX_SPAWN_SETSCHEDPARAM       0x10
#define POSIX_SPAWN_SETSCHEDULER        0x20

typedef struct {
    int __allocated;
    int __used;
    struct __spawn_action *__actions;
    int __pad[16];
} posix_spawn_file_actions_t;
typedef struct {
    short __flags;
    pid_t __pgid;
    sigset_t __sd;
    sigset_t __ss;
    struct sched_param __sp;
    int __policy;
    int __pad[16];
} posix_spawnattr_t;

extern int posix_spawn(pid_t *__pid, const char *__path,
                        const posix_spawn_file_actions_t *__file_actions,
                        const posix_spawnattr_t *__attrp,
                        char *const argv[], char *const envp[]);

extern int
posix_spawn_file_actions_addclose(posix_spawn_file_actions_t * __file_actions, int __fd);
extern int posix_spawn_file_actions_adddup2(posix_spawn_file_actions_t * __file_actions, int __fd, int __newfd);
extern int posix_spawn_file_actions_addopen(posix_spawn_file_actions_t * __file_actions, int __fd, const char *__path, int __oflag, mode_t __mode);
extern int posix_spawn_file_actions_destroy(posix_spawn_file_actions_t * __file_actions);
extern int posix_spawn_file_actions_init(posix_spawn_file_actions_t * __file_actions);
extern int posix_spawnattr_destroy(posix_spawnattr_t * __attr);
extern int posix_spawnattr_getflags(const posix_spawnattr_t * __attr, short int *__flags);
extern int posix_spawnattr_getpgroup(const posix_spawnattr_t * __attr, pid_t *__pgroup);
extern int posix_spawnattr_getschedparam(const posix_spawnattr_t * __attr, struct sched_param *__schedparam);
extern int posix_spawnattr_getschedpolicy(const posix_spawnattr_t * __attr, int *__schedpolicy);
extern int posix_spawnattr_getsigdefault(const posix_spawnattr_t * __attr, sigset_t *__sigdefault);
extern int posix_spawnattr_getsigmask(const posix_spawnattr_t * __attr, sigset_t *__sigmask);
extern int posix_spawnattr_init(posix_spawnattr_t * __attr);
extern int posix_spawnattr_setflags(posix_spawnattr_t * __attr, short int __flags);
extern int posix_spawnattr_setpgroup(posix_spawnattr_t * __attr, pid_t __pgroup);
extern int posix_spawnattr_setschedparam(posix_spawnattr_t * __attr, const struct sched_param * __schedparam);
extern int posix_spawnattr_setschedpolicy(posix_spawnattr_t * __attr, int __schedpolicy);
extern int posix_spawnattr_setsigdefault(posix_spawnattr_t * __attr, const sigset_t * __sigdefault);
extern int posix_spawnattr_setsigmask(posix_spawnattr_t * __attr, const sigset_t * __sigmask);

extern int posix_spawnp(pid_t *__pid, const char *__file, const posix_spawn_file_actions_t *__file_actions, const posix_spawnattr_t *__attrp, char *const argv[], char *const envp[]);
14.4.62 stddef.h

```c
#ifdef !defined(__GNUC__)
#define  __builtin_offsetof  (TYPE,  MEMBER)
    ((size_t)&((TYPE*)0)->MEMBER)
#endif
#ifndef NULL
#ifdef __cplusplus
#define NULL        (0L)
#else
#define NULL        ((void*) 0)
#endif
#endif
#define offsetof(TYPE,MEMBER)   __builtin_offsetof (TYPE, MEMBER)
```

14.4.63 stdint.h

```c
#define INT16_C(c)      c
#define INT32_C(c)      c
#define INT8_C(c)       c
#define UINT16_C(c)     c
#define UINT8_C(c)      c
#define UINT32_C(c)     c
#define INT8_MIN        (-128)
#define INT_FAST8_MIN   (-128)
#define INT_LEAST8_MIN  (-128)
#define INT32_MIN       (-2147483647-1)
#define INT_LEAST32_MIN (-2147483647-1)
#define SIG_ATOMIC_MIN  (-2147483647-1)
#define INT16_MIN       (-32767-1)
#define INT_LEAST16_MIN (-32767-1)
#define INT64_MIN       (__INT64_C(9223372036854775807)-1)
#define INTMAX_MIN      (__INT64_C(9223372036854775807)-1)
#define INT_FAST64_MIN  (__INT64_C(9223372036854775807)-1)
#define INT_LEAST64_MIN (__INT64_C(9223372036854775807)-1)
#define WINT_MIN        (0u)
#define INT8_MAX        (127)
#define INT_FAST8_MAX   (127)
#define INT_LEAST8_MAX  (127)
#define INT32_MAX       (2147483647)
#define INT_LEAST32_MAX (2147483647)
#define SIG_ATOMIC_MAX  (2147483647)
#define UINT8_MAX       (255)
#define UINT_FAST8_MAX  (255)
#define UINT_LEAST8_MAX (255)
#define INT16_MAX       (32767)
#define INT_LEAST16_MAX (32767)
#define INT64_MAX       (__INT64_C(9223372036854775807))
#define INTMAX_MAX      (__INT64_C(9223372036854775807))
#define INT_FAST64_MAX  (__INT64_C(9223372036854775807))
#define INT_LEAST64_MAX (__INT64_C(9223372036854775807))
#define UINT64_MAX      (__UINT64_C(18446744073709551615))
#define UINTMAX_MAX     (__UINT64_C(18446744073709551615))
#define UINT_FAST64_MAX (__UINT64_C(18446744073709551615))
```

© 2015 Linux Foundation
typedef signed char int8_t;
typedef short int16_t;
typedef int int32_t;
typedef unsigned char uint8_t;
typedef unsigned short uint16_t;
typedef unsigned int uint32_t;
typedef signed char int_least8_t;
typedef short int int_least16_t;
typedef int int_least32_t;
typedef unsigned char uint_least8_t;
typedef unsigned short uint_least16_t;
typedef unsigned int uint_least32_t;
typedef signed char int_fast8_t;
typedef unsigned char uint_fast8_t;

14.4.64 stdio.h

#define EOF     (-1)
#define P_tmpdir       "/tmp"
#ifndef SEEK_SET
#define SEEK_SET        0
#endif
#else
#define SEEK_CUR        1
#endif
#ifndef SEEK_END
#define SEEK_END        2
#endif
#define FOPEN_MAX       16
#define L_tmpnam        20
#define TMP_MAX 238328
#define FILENAME_MAX    4096
#define BUFSIZ  8192
#define L_ctermid       9
#define L_cuserid       9

typedef struct {
    off_t __pos;
    mbstate_t __state;
} fpos_t;
typedef struct {
    off64_t __pos;
    mbstate_t __state;
} fpos64_t;
typedef struct _IO_FILE FILE;

#define _IOFBF  0
#define _IOLBF  1
#define _IONBF  2

extern char *__fgets_chk(char *, size_t, int, FILE *);
extern char *__fgets_unlocked_chk(char *, size_t, int, FILE *);
extern size_t __fpending(FILE *);
extern int __fprintf_chk(FILE *, int, const char *, ...);
extern int __printf_chk(int, const char *, ...);
extern int __snprintf_chk(char *, size_t, int, size_t, const char *, ...);
extern int __sprintf_chk(char *, int, size_t, const char *, ...);
extern int __vfprintf_chk(FILE *, int, const char *, va_list);
extern int __vprintf_chk(int, const char *, va_list);
extern int __vsnprintf_chk(char *, size_t, int, size_t, const char *, va_list);
extern int __vsprintf_chk(char *, int, size_t, const char *, ...);
va_list);
extern char *const _sys_errlist[];
extern int asprintf(char **__ptr, const char *__fmt, ...);
extern void clearerr(FILE *__stream);
extern int dprintf(int __fd, const char *__fmt, ...);
extern FILE *fdopen(int __fd, const char *__modes);
extern int feof(FILE *__stream);
extern int feof_unlocked(FILE *__stream);
extern int ferr(FIELD *__stream);
extern int ferr_unlocked(FILE *__stream);
extern int fflush(FILE *__stream);
extern int fflush_unlocked(FILE *__stream);
extern int fgetc(FILE *__stream);
extern int fgetc_unlocked(FILE *__stream);
extern int fgets(char *__s, int __n, FILE *__stream);
extern int fgets_unlocked(char *__s, int __n, FILE *__stream);
extern int fileno(FIELD *__stream);
extern int fileno_unlocked(FILE *__stream);
extern int ferror(FILE *__stream);
extern int ferror_unlocked(FILE *__stream);
extern int fflush(FILE *__stream);
extern int fflush_unlocked(FILE *__stream);
extern int fgetc(FILE *__stream);
extern int fgetpos(FILE *__stream, fpos_t *__pos);
extern int fgetpos64(FILE *__stream, fpos64_t *__pos);
extern int fgetc_unlocked(FILE *__stream, const char *__fmt, ...);
extern int fnpos(char *__s, int __n, FILE *__stream);
extern int fnpos_unlocked(FILE *__stream, const char *__fmt, ...);
extern int fprintf(FILE *__stream, const char *__format, ...);
extern int fprintf_unlocked(const char *__s, size_t __len, const char * __fmt, ...);
extern int fputc(int __c, FILE *__stream);
extern int fputc_unlocked(int __c, FILE *__stream);
extern int fputs(const char *__s, FILE *__stream);
extern int fputs_unlocked(const char *__s, FILE *__stream);
extern size_t fread(void *__ptr, size_t __size, size_t __n, FILE *__stream);
extern size_t fread_unlocked(void *__ptr, size_t __size, size_t __n, FILE *__stream);
extern size_t fwrite(const void *__ptr, size_t __size, size_t __n, FILE *__stream);
extern size_t fwrite_unlocked(const void *__ptr, size_t __size, size_t __n, FILE *__stream);
 extern int getc(FILE *__stream);
 extern int getc_unlocked(FILE *__stream);
 extern int getchar(void);
 extern int getchar_unlocked(void);
 extern ssize_t getdelim(char **__lineptr, size_t __n, int __delimiter,
```c
FILE *__stream);
extern ssize_t getline(char **__lineptr, size_t *__n, FILE *__stream);
extern int getw(FILE *__stream);
extern FILE *open_memstream(char **__bufloc, size_t *__sizeloc);
extern int pclose(FILE *__stream);
extern void perror(const char *__s);
extern FILE *popen(const char *__command, const char *__modes);
extern int printf(const char *__format, ...);
extern int putc(int __c, FILE *__stream);
extern int putc_unlocked(int __c, FILE *__stream);
extern int putchar(int __c);
extern int putchar_unlocked(int __c);
extern int puts(const char *__s);
extern int putw(int __w, FILE *__stream);
extern int remove(const char *__filename);
extern int rename(const char *__old, const char *__new);
extern int renameat(int __oldfd, const char *__old, int __newfd,
        const char *__new);
extern void rewind(FILE *__stream);
extern int scanf(const char *__format, ...);
extern void setbuf(FILE *__stream, char *__buf);
extern void setbuffer(FILE *__stream, char *__buf, size_t __size);
extern int setvbuf(FILE *__stream, char *__buf, int __modes,
        size_t __n);
extern int snprintf(char *__s, size_t __maxlen, const char *__format, ...);
extern int sprintf(char *__s, const char *__format, ...);
extern int sscanf(const char *__s, const char *__format, ...);
extern FILE *stderr;
extern FILE *stdin;
extern FILE *stdout;
extern char *tempnam(const char *__dir, const char *__pfx);
extern FILE *tmpfile(void);
extern FILE *tmpfile64(void);
extern char *tmpnam(char *__s);
extern int ungetc(int __c, FILE *__stream);
extern int vasprintf(char **__ptr, const char *__f, va_list __arg);
extern int vdprintf(int __fd, const char *__fmt, va_list __arg);
extern int vfprintf(FILE *__s, const char *__format, va_list __arg);
extern int vfscanf(FILE *__s, const char *__format, va_list __arg);
extern int vprintf(const char *__format, va_list __arg);
extern int vscanf(const char *__format, va_list __arg);
extern int vsnprintf(char *__s, size_t __maxlen, const char *__format,
        va_list __arg);
extern int vsprintf(char *__s, const char *__format, va_list __arg);
extern int vsscanf(const char *__s, const char *__format, va_list __arg);

14.4.65 stdlib.h

#define MB_CUR_MAX  (__ctype_get_mb_cur_max())
#define EXIT_SUCCESS 0
#define EXIT_FAILURE 1
#define RAND_MAX 2147483647

struct drand48_data {
    unsigned short __x[3];
    unsigned short __old_x[3];
}
```
unsigned short __c;
unsigned short __init;
unsigned long long int __a;
};
typedef int (*__compar_fn_t) (const void *, const void *);

struct random_data {
    int32_t *fptr;              /* Front pointer. */
    int32_t *rptr;              /* Rear pointer. */
    int32_t *state;             /* Array of state values. */
    int rand_type;              /* Type of random number generator. */
    int rand_deg;               /* Degree of random number generator. */
    int rand_sep;               /* Distance between front and rear. */
    int32_t *end_ptr;           /* Pointer behind state table. */
};
typedef struct {
    int quot;
    int rem;
} div_t;

typedef struct {
    long int quot;
    long int rem;
} ldiv_t;

typedef struct {
    long long int quot;
    long long int rem;
} lldiv_t;

extern void _Exit(int __status);
extern size_t __ctype_get_mb_cur_max(void);
extern size_t __mbstowcs_chk(wchar_t *, const char *, size_t, size_t);
extern char *__realpath_chk(const char *, char *, size_t);
extern double __strtod_internal(const char *, char **, int);
extern float __strtof_internal(const char *, char **, int);
extern long int __strtol_internal(const char *, char **, int);
extern long double __strtold_internal(const char *, char **, int);
extern long long int __strtoll_internal(const char *, char **, int);
extern unsigned long int __strtol_internal(const char *, char **, int);
extern size_t __wcstombs_chk(char *, const wchar_t *, size_t, size_t);
extern int __wctomb_chk(char *, wchar_t, size_t);
extern long int a64l(const char *__s);
extern void abort(void);
extern int abs(int __x);
extern int atexit(void (*__func) (void));
extern double atof(const char *__nptr);
extern int atoi(const char *__nptr);
extern long int atol(const char *__nptr);
extern long long int atoll(const char *__nptr);
extern void *bsearch(const void * __key, const void * __base, size_t __nmemb,
    size_t __size, __compar_fn_t __compar);
extern void *calloc(size_t __nmemb, size_t __size);
extern div_t div(int __numer, int __denom);
extern double drand48(void);
extern int drand48_r(struct drand48_data *buffer, double *result);
extern char *ecvt(double __value, int __ndigit, int *__decpt, int *__sign);
extern char **environ;
extern double drand48(unsigned short __xsubi[3]);
extern int erand48_r(unsigned short __xsubi[3],
                     struct drand48_data *buffer, double *result);
extern void exit(int __status);
extern char *fcvt(double __value, int __ndigit, int *__decpt, int *__sign);
extern void free(void *__ptr);
extern char *gcvt(double __value, int __ndigit, char *__buf);
extern char *getenv(const char *__name);
extern int getloadavg(double __loadavg[], int __nelem);
extern int getsubopt(char **optionp, char *const *__tokens,
                      char **valuep);
extern int grantpt(int __fd);
extern char *initstate(unsigned int __seed, char *__statebuf,
                       size_t __statelen);
extern int initstate_r(unsigned int __seed, char *__statebuf,
                       size_t __statelen, struct random_data *buf);
extern long int jrand48(unsigned short __xsubi[3]);
extern int jrand48_r(unsigned short __xsubi[3],
                     struct drand48_data *__buffer, long int *__result);
extern char *l64a(long int __n);
extern long int labs(long int __x);
extern void lcong48(unsigned short __param[7]);
extern int lcong48_r(unsigned short __param[7],
                     struct drand48_data *__buffer);
extern ldiv_t ldiv(long int __numer, long int __denom);
extern long long int llabs(long long int __x);
extern lldiv_t lldiv(long long int __numer, long long int __denom);
extern long int lrand48(void);
extern int lrand48_r(struct drand48_data *buffer, long int *__result);
extern void *malloc(size_t __size);
extern int mblen(const char *__s, size_t __n);
extern size_t mbstowcs(wchar_t * __pwcs, const char *__s, size_t __n);
extern int mbtowc(wchar_t * __pwc, const char *__s, size_t __n);
extern char *mkdtemp(char *__template);
extern int mkstemp(char *__template);
extern int mkstemp64(char *__template);
extern char *mktemp(char *__template);
extern long int mrand48(void);
extern int mrand48_r(struct drand48_data *buffer, long int *__result);
extern long int mrand48(unsigned short __xsubi[3]);
extern int mrand48_r(unsigned short __xsubi[3],
                     struct drand48_data *buffer, long int *__result);
extern int posix_memalign(void **memptr, size_t __alignment, size_t __size);
extern int posix_openpt(int __oflag);
extern char *ptsname(int __fd);
extern int putenv(char *__string);
extern void qsort(void *__base, size_t __nmemb, size_t __size,
                  const __compar_fn_t __compar);
extern int rand(void);
extern int rand_r(unsigned int *__seed);
extern long int random(void);
extern int random_r(struct random_data *__buf, int32_t *__result);
extern void *realloc(void *__ptr, size_t __size);
extern char *realpath(const char *__name, char *__resolved);
extern unsigned short *seed48(unsigned short __seed16v[3]);
extern int seed48_r(unsigned short __seed16v[3],
                      struct drand48_data *__buffer);
extern int setenv(const char *__name, const char *__value, int __replace);
extern char *setstate(char *__statebuf);
extern int setstate_r(char *__statebuf, struct random_data *__buf);
extern void srand(unsigned int __seed);
extern void srand48(long int __seedval);
extern int srand48_r(long int __seedval, struct drand48_data *__buffer);
extern void srandom(unsigned int __seed);
extern int srandom_r(unsigned int __seed, struct random_data *__buf);
extern double strtod(const char *__nptr, char **__endptr);
extern float strtof(const char *__nptr, char **__endptr);
extern long int strtol(const char *__nptr, char **__endptr, int __base);
extern long double strtold(const char *__nptr, char **__endptr);
extern long long int strtoll(const char *__nptr, char **__endptr,
                                      int __base);
extern long long int strtoq(const char *__nptr, char **__endptr,
                                      int __base);
extern unsigned long int strtoul(const char *__nptr, char **__endptr,
                                      int __base);
extern unsigned long int strtoull(const char *__nptr, char **__endptr,
                                      int __base);
extern unsigned long int strtoq(const char *__nptr, char **__endptr,
                                      int __base);
extern int system(const char *__command);
extern int unlockpt(int __fd);
extern int unsetenv(const char *__name);
extern size_t wcstombs(char *__s, const wchar_t *__pwcs, size_t __n);
extern int wctomb(char *__s, wchar_t __wchar);

14.4.66 string.h

#define strftime_r __xpg_strerror_r
#define bzero(s,n) memset(s,0,n)

extern void *memcpy_chk(void *, const void *, size_t, size_t);
extern void *memmove_chk(void *, const void *, size_t, size_t);
extern void *memcpyp(void *dest, const void *src, size_t __n);
extern void *memcpy_chk(void *, const void *, size_t, size_t);
extern int memmove_chk(void *, const void *, size_t, size_t);
extern char *memset_chk(void *, int, size_t, size_t);
extern char *strncpy_chk(char *, const char *, size_t, size_t);
extern char *strncpy_chk(char *, const char *, size_t, size_t);
extern char *strcat_chk(char *, const char *, size_t);
extern char *strcatcp(char *, const char *, size_t);
extern char *strncat_chk(char *, const char *, size_t, size_t);
extern char *strncpyc_chk(char *, const char *, size_t, size_t);
14.4.67 strings.h

extern int bcmp(const void *__s1, const void *__s2, size_t __n);
extern void bcopy(const void *__src, void *__dest, size_t __n);
extern void bzero(void *__s, size_t __n);
extern int ffs(int __i);
extern char *index(const char *__s, int __c);
extern char *rindex(const char *__s, int __c);
extern int strcasecmp(const char *__s1, const char *__s2);
extern int strcasecmp_l(const char *s1, const char *s2, locale_t locale);
extern char *strcpyp(const char *__s, const char *__t);
extern char *strcat(char *__s, const char *__t);
extern char *strchr(const char *__s, int __c);
extern int strcmp(const char *__s1, const char *__s2);
extern int strcoll(const char *__s1, const char *__s2);
extern int strcoll_l(const char *s1, const char *s2, locale_t locale);
extern char *strcpy(char *__s, const char *__t);
extern char *strdup(const char *__s);
extern char *strerror(int __errnum);
extern char *strerror_l(int __errnum, locale_t locale);
extern size_t strlen(const char *__s);
extern char *strcasecmp(const char *__s1, const char *__s2);
extern char *strcasecmp(const char *__s1, const char *__s2, locale_t locale);
locale);
extern int strncasecmp(const char *__s1, const char *__s2, size_t __n);
extern int strncasecmp_l(const char *__s1, const char *__s2, size_t __n,
    locale_t locale);

14.4.68 sys/epoll.h

#define EPOLL_CTL_ADD   1       /* Add a file descriptor to the interface. */
#define EPOLL_CTL_DEL   2       /* Remove a file descriptor from the interface. */
#define EPOLL_CTL_MOD   3       /* Change file descriptor epoll_event structure. */
#define EPOLLIN 1
#define EPOLLPRI 2
#define EPOLLOUT 4
#define EPOLLERR 8
#define EPOLLHUP 16
#define EPOLLRDHUP 0x2000
#define EPOLLONESHOT (1 << 30)
#define EPOLLET (1 << 31)
typedef union epoll_data {
    void *ptr;
    int fd;
    uint32_t u32;
    uint64_t u64;
} epoll_data_t;

struct epoll_event {
    uint32_t events;
    epoll_data_t data;
};

extern int epoll_create(int __size);
extern int epoll_ctl(int __epfd, int __op, int __fd,
                      struct epoll_event *__event);
extern int epoll_wait(int __epfd, struct epoll_event *__events,
                       int __maxevents, int __timeout);

14.4.69 sys/file.h

#define LOCK_SH 1
#define LOCK_EX 2
#define LOCK_NB 4
#define LOCK_UN 8

extern int flock(int __fd, int __operation);

14.4.70 sys/inotify.h

#define IN_ACCESS 0x00000001
#define IN_MODIFY 0x00000002
#define IN_ATTRIB 0x00000004
#define IN_CLOSE_WRITE 0x00000008
#define IN_CLOSE_NOWRITE 0x00000010
#define IN_OPEN 0x00000020
#define IN_MOVED_FROM 0x00000040
#define IN_MOVED_TO 0x00000080
#define IN_CREATE 0x00000100
#define IN_DELETE 0x00000200
#define IN_DELETE_SELF 0x00000400
#define IN_MOVE_SELF 0x00000800
#define IN_UNMOUNT 0x00002000
#define IN_Q_OVERFLOW 0x00004000
#define IN_IGNORED 0x00008000
#define IN_ISDIR 0x40000000
#define IN_ONESHOT 0x80000000
#define IN_CLOSE (IN_CLOSE_WRITE | IN_CLOSE_NOWRITE)
#define IN_MOVE (IN_MOVED_FROM | IN_MOVED_TO)
#define IN_ALL_EVENTS \  
  (IN_ACCESS | IN_MODIFY | IN_ATTRIB | IN_CLOSE_WRITE | \ 
   IN_CLOSE_NOWRITE | IN_OPEN | IN_MOVED_FROM | IN_MOVED_TO | 
   IN_CREATE | \ 
   IN_DELETE | IN_DELETE_SELF | IN_MOVE_SELF)

struct inotify_event {
  int wd;
  uint32_t mask;
  uint32_t cookie;
  uint32_t len;
  char name[];
};

extern int inotify_add_watch(int __fd, const char *__name,
  uint32_t __mask);

extern int inotify_init(void);

extern int inotify_rm_watch(int __fd, int __wd);

14.4.71 sys/ioctl.h

#define _IOC(dir,type,nr,size)   (((dir)  <<  _IOC_DIRSHIFT)  | 
  ((type) << _IOC_TYPESHIFT) | ((nr) << _IOC_NRSHIFT) | ((size) << 
  _IOC_SIZESHIFT))
#define _IOC_DIR(nr)    (((nr) >> _IOC_DIRSHIFT) & _IOC_DIRMASK)
#define _IOC_NR(nr)     (((nr) >> _IOC_NRSHIFT) & _IOC_NRMASK)
#define  _IOC_SIZE(nr)    (((nr)  >>  _IOC_SIZESHIFT)  & 
  _IOC_SIZEMASK)
#define  _IOC_TYPE(nr)    (((nr)  >>  _IOC_TYPESHIFT)  & 
  _IOC_TYPEMASK)
#define _IOC_DIRMASK    ((1 << _IOC_DIRBITS)-1)
#define _IOC_NRMASK     ((1 << _IOC_NRBITS)-1)
#define _IOC_SIZEMASK   ((1 << _IOC_SIZEBITS)-1)
#define _IOC_TYPEMASK   ((1 << _IOC_TYPEBITS)-1)
#define IOCT_INOUT       ((_IOC_WRITE|_IOC_READ) << _IOC_DIRSHIFT)
#define _IOC_TYPECHECK(t)       (sizeof(t))
#define _IOC_TYPESHIFT  (_IOC_NRSHIFT+_IOC_NRBITS)
#define IOC_OUT (_IOC_READ << _IOC_DIRSHIFT)
#define IOCSIZE_MASK    (_IOC_SIZEMASK << _IOC_SIZESHIFT)
#define IOCSIZE_SHIFT   (_IOC_SIZESHIFT)
#define _IOC_DIRSHIFT   (_IOC_SIZESHIFT+_IOC_SIZEBITS)
#define IOC_IN  (_IOC_WRITE << _IOC_DIRSHIFT)
#define _IOC_NRSHIFT    0
#define _IOC_NONE       0U
#define _IOC_SIZEBITS   14
#define _IOC_WRITE      1U
#define _IOC_DIRBITS    2
#define _IOC_READ       2U
#define _IOC_NRBITS     8
#define _IOC_TYPEBITS   8
#define _IO(type,nr)     _IOC(_IOC_NONE,(type),(nr),0)
#define _IOR(type,nr,size)    _IOC(_IOC_READ,(type),(nr), 
  (sizeof(t)))
#define _IOR_BAD(type,nr,size)    _IOC(_IOC_READ,(type), 
  (nr),sizeof(size))
#define _IOWR(type,nr,size)     _IOC(_IOC_READ|_IOC_WRITE,(type),
#define _IOWR_BAD(type,nr,size) _IOC(_IOC_READ|_IOC_WRITE,(type),(nr),sizeof(size))
#define _IOW(type,nr,size) _IOC(_IOC_WRITE,(type),(nr),sizeof(size))
#define  _IOW_BAD(type,nr,size)   _IOC(_IOC_WRITE,(type),(nr),sizeof(size))

struct winsize {
    unsigned short ws_row;      /* Rows, in characters. */
    unsigned short ws_col;      /* Columns, in characters. */
    unsigned short ws_xpixel;   /* Horizontal pixels. */
    unsigned short ws_ypixel;   /* Vertical pixels. */
};

extern int ioctl(int __fd, unsigned long int __request, ...);

14.4.72 sys/ipc.h
#define IPC_PRIVATE     ((key_t)0)
#define IPC_RMID        0
#define IPC_CREAT       00001000
#define IPC_EXCL        00002000
#define IPC_NOWAIT      00004000
#define  IPC_SET 1
#define  IPC_STAT        2

extern key_t ftok(const char *__pathname, int __proj_id);

14.4.73 sys/mman.h
#define MAP_FAILED      ((void*)-1)
#define POSIX_MADV_NORMAL       0
#define PROT_NONE       0x0
#define MAP_SHARED      0x01
#define MAP_PRIVATE     0x02
#define PROT_READ       0x1
#define MAP_FIXED       0x10
#define PROT_WRITE      0x2
#define MAP_ANONYMOUS   0x20
#define PROT_EXEC       0x4
#define MREMAP_MAYMOVE  1
#define MS_ASYNC        1
#define POSIX_MADV_RANDOM       1
#define MREMAP_FIXED    2
#define MS_INVALIDATE   2
#define POSIX_MADV_SEQUENTIAL   2
#define POSIX_MADV_WILLNEED     3
#define MS_SYNC 4
#define POSIX_MADV_DONTNEED     4
#define MAP_ANON        MAP_ANONYMOUS

extern int mlock(const void *__addr, size_t __len);
extern int mlockall(int __flags);
extern void *mmap(void *__addr, size_t __len, int __prot, int __flags, int __fd, off_t __offset);
extern void *mmap64(void *__addr, size_t __len, int __prot, int __flags, int __fd, off64_t __offset);
extern int mprotect(void *__addr, size_t __len, int __prot);
extern void *mremap(void *__addr, size_t __len, size_t __new_len, int __flags, ...);
extern int msync(void *__addr, size_t __len, int __flags);
extern int munlock(const void *__addr, size_t __len);
extern int munlockall(void);
extern int munmap(void *__addr, size_t __len);
extern int posix_madvise(void *__addr, size_t __len, int __advice);
extern int shm_open(const char *__name, int __oflag, mode_t __mode);
extern int shm_unlink(const char *__name);

14.4.74 sys/msg.h

#define MSG_NOERROR 010000
extern int msgctl(int __msqid, int __cmd, struct msqid_ds *__buf);
extern int msgget(key_t __key, int __msgflg);
extern ssize_t msgrcv(int __msqid, void *__msgp, size_t __msgsz,
                       long int __msgtyp, int __msgflg);
extern int msgsnd(int __msqid, const void *__msgp, size_t __msgsz,
                       int __msgflg);

14.4.75 sys/param.h

#define NOFILE 256
#define MAXPATHLEN 4096

14.4.76 sys/poll.h

#define POLLIN 0x0001 /* There is data to read */
#define POLLPRI 0x0002 /* There is urgent data to read */
#define POLLOUT 0x0004 /* Writing now will not block */
#define POLLERR 0x0008 /* Error condition */
#define POLLHUP 0x0010 /* Hung up */
#define POLLNVAL 0x0020 /* Invalid request: fd not open */
#define POLLRDNORM 0x0040 /* Normal data may be read */
#define POLLRDBAND 0x0080 /* Priority data may be read */
#define POLLWRNORM 0x0100 /* Writing now will not block */
#define POLLWRBAND 0x0200 /* Priority data may be written */

struct pollfd {
    int fd;          /* File descriptor to poll. */
    short events;   /* Types of events poller cares about. */
    short revents;  /* Types of events that actually occurred. */
};
typedef unsigned long int nfds_t;

14.4.77 sys/ptrace.h

enum __ptrace_setoptions {
    PTRACE_O_TRACESYSGOOD = 0x00000001,
    PTRACE_O_TRACEFORK = 0x00000002,
    PTRACE_O_TRACEVFORK = 0x00000004,
    PTRACE_O_TRACECLONE = 0x00000008,
};
LSB Core - Generic 5.0

PTRACE_O_TRACEEXEC = 0x00000010,
PTRACE_O_TRACEVFORKDONE = 0x00000020,
PTRACE_O_TRACEEXIT = 0x00000040,
PTRACE_O_MASK = 0x00000007f
};

enum __ptrace_eventcodes {
    PTRACE_EVENT_FORK = 1,
    PTRACE_EVENT_VFORK = 2,
    PTRACE_EVENT_CLONE = 3,
    PTRACE_EVENT_EXEC = 4,
    PTRACE_EVENT_VFORK_DONE = 5,
    PTRACE_EVENT_EXIT = 6
};

extern long int ptrace(enum __ptrace_request, ...);

14.4.78 sys/resource.h

#define RUSAGE_CHILDREN (-1)
#define RLIM_INFINITY (-0UL)
#define RLIM_SAVED_CUR -1
#define RLIM_SAVED_MAX -1
#define RLIMIT_CPU 0
#define RUSAGE_SELF 0
#define RLIMITFSIZE 1
#define RLIMIT_LOCKS 10
#define RLIMIT_SIGPENDING 11
#define RLIMIT_MSGQUEUE 12
#define RLIMIT_NICE 13
#define RLIMIT_RTPRIO 14
#define RLIMIT_RTTIME 15
#define RLIMIT_NLIMITS 16
#define RLIMIT_DATA 2
#define RLIMIT_STACK 3
#define RLIMIT_CORE 4
#define RLIMIT_RSS 5
#define RLIMIT_NPROC 6
#define RLIMIT_NOFILE 7
#define RLIMIT_MEMLOCK 8
#define RLIMIT_AS 9

typedef unsigned long int rlim_t;
typedef unsigned long int rlim64_t;
typedef int __rlimit_resource_t;

struct rlimit {
    rlim_t rlim_cur;            /* The current (soft) limit. */
    rlim_t rlim_max;            /* The hard limit. */
};

struct rlimit64 {
    rlim64_t rlim_cur;          /* The current (soft) limit. */
    rlim64_t rlim_max;          /* The hard limit. */
};

struct rusage {
    struct timeval ru_utime;    /* Total amount of user time
used. */
    struct timeval ru_stime;    /* Total amount of system time
used. */
    long int ru_maxrss;         /* Maximum resident set size (in
kilobytes). */
    long int ru_ixrss;          /* Amount of sharing of text
segment memory with other p. */
    long int ru_idrss;          /* Amount of data segment memory
used (kilobyte-seconds). */
    long int ru_isrss;          /* Amount of stack memory used..."}
long int ru_minflt; /* Number of soft page faults 
   (i.e. those serviced by reclaimin */
long int ru_majflt; /* Number of hard page faults 
   (i.e. those that required I/O). */
long int ru_nswap; /* Number of times a process was 
swapped out of physical memory */
long int ru_inblock; /* Number of input operations via 
the file system. Note: This */
long int ru_oublock; /* Number of output operations 
via the file system. */
long int ru_msgsnd; /* Number of IPC messages sent. */
long int ru_msgrcv; /* Number of IPC messages 
received. */
long int ru_nsignals; /* Number of signals delivered. */
long int ru_nvcsw; /* Number of voluntary context 
switches, i.e. because the proce */
long int ru_nivcsw; /* Number of involuntary context 
switches, i.e. a higher priori */
};

enum __priority_which {
   PRIO_PROCESS = 0, /* WHO is a process ID. */
   PRIO_PGRP = 1, /* WHO is a process group ID. */
   PRIO_USER = 2 /* WHO is a user ID. */
};
#define PRIO_PGRP       PRIO_PGRP
#define PRIO_PROCESS    PRIO_PROCESS
#define PRIO_USER       PRIO_USER

typedef enum __priority_which __priority_which_t;
extern int getpriority(__priority_which_t __which, id_t __who);
extern int getrlimit(__rlimit_resource_t __resource,
   struct rlimit *__rlimits);
extern int getrlimit64(__rlimit_resource_t __resource,
   struct rlimit64 *__rlimits);
extern int getrusage(int __who, struct rusage *__usage);
extern int setpriority(__priority_which_t __which, id_t __who, 
   int __prio);
extern int setrlimit(__rlimit_resource_t __resource, 
   const struct rlimit *__rlimits);
extern int setrlimit64(__rlimit_resource_t __resource, 
   const struct rlimit64 *__rlimits);

14.4.79 sys/select.h

define FD_ISSET(d,set) (((set)->fds_bits[(d)/(8*sizeof(long)))&(1L<<(d)%(8*sizeof(long)))]!=0)
define FD_CLR(d,set) (((set)->fds_bits[(d)/(8*sizeof(long))])&=(1L<<((d)%(8*sizeof(long)))))
define FD_SET(d,set) (((set)->fds_bits[(d)/(8*sizeof(long))])=(1L<<((d)%(8*sizeof(long)))))
define NFDBITS (8 * sizeof(long))
define FD_SETSIZE 1024
#define FD_ZERO(fdsetp) bzero(fdsetp, sizeof(*fdsetp))

typedef struct {
   unsigned long int fds_bits[FD_SETSIZE / NFDBITS];
} fd_set;
extern int pselect(int __nfds, fd_set * __readfds, fd_set * 
   __writefds, fd_set * __exceptfds, const struct timespec
*__timeout,
    const sigset_t *__sigmask);
extern int select(int __nfds, fd_set *__readfds, fd_set *__writefds,
    fd_set *__exceptfds, struct timeval *__timeout);

14.4.80 sys/sem.h

#define SEM_UNDO        0x1000
#define GETPID  11
#define GETVAL  12
#define GETALL  13
#define GETNCNT 14
#define GETZCNT 15
#define SETVAL  16
#define SETALL  17
struct sembuf {
    short sem_num;
    short sem_op;
    short sem_flg;
};
extern int semctl(int __semid, int __semnum, int __cmd, ...);
extern int semget(key_t __key, int __nsems, int __semflg);
extern int semop(int __semid, struct sembuf *__sops, size_t __nsops);

14.4.81 sys/sendfile.h

extern ssize_t sendfile(int __out_fd, int __in_fd, off_t *__offset,
    size_t __count);
extern ssize_t sendfile64(int __out_fd, int __in_fd, off64_t *__offset,
    size_t __count);

14.4.82 sys/shm.h

#define SHM_RDONLY      010000
#define SHM_W   0200
#define SHM_RND 020000
#define SHM_R   0400
#define SHM_REMAP       040000
#define SHM_LOCK        11
#define SHM_UNLOCK      12
extern int __getpagesize(void);
extern void *shmat(int __shmid, const void *__shmaddr, int __shmid);
extern int shmctl(int __shmid, int __cmd, struct shmid_ds *__buf);
extern int shmdt(const void *__shmaddr);
extern int shmget(key_t __key, size_t __size, int __shmflg);

14.4.83 sys/socket.h

#define CMSG_FIRSTHDR(msg) ((size_t) (msg)->msg_controllen
    >= sizeof(struct cmsghdr) ? (struct cmsghdr *)(msg)-
    >msg_control : (struct cmsghdr *) NULL)
#define CMSG_LEN(len)    (CMSG_ALIGN(sizeof(struct cmsghdr))+(len))
#define SCM_RIGHTS      0x01
#define SOL_SOCKET      1
#define SOMAXCONN       128
#define SOL_RAW 255
#define CMSG_ALIGN(len) \((len)+sizeof(size_t)-1)&(size_t)-sizeof(size_t)-1\)
#define CMSG_DATA(cmsg) \((unsigned char *) (cmsg) + CMSG_ALIGN(sizeof(struct cmsghdr)))\)
#define CMSG_SPACE(len) \(CMSG_ALIGN(sizeof(struct cmsghdr))+CMSG_ALIGN(len)\)
#define CMSG_NXTHDR(mhdr,cmsg)  \(((cmsg) == NULL) ? CMSG_FIRSTHDR(mhdr) : \(((u_char *)cmsg) + CMSG_ALIGN((cmsg)->cmsg_len) \+
CMSG_ALIGN(sizeof(struct cmsghdr)))\)

struct linger {  
    int l_onoff;  
    int l_linger;  
};
struct cmsghdr {  
    size_t cmsg_len;  
    int cmsg_level;  
    int cmsg_type;  
};
struct iovec {  
    void *iov_base;  
    size_t iov_len;  
};
typedef unsigned short sa_family_t;  
typedef unsigned int socklen_t;
struct sockaddr {  
    sa_family_t sa_family;  
    char sa_data[14];  
};
struct sockaddr_storage {  
    sa_family_t ss_family;  
    __ss_align type __ss_align;  
    char __ss_padding[(128 - (2 * sizeof(__ss_align type)))];  
};
struct msghdr {  
    void *msg_name;  
    int msg_name len;  
    struct iovec *msg_irov;  
    size_t msg iovlen;  
    void *msg_control;  
    size_t msg control len;  
    unsigned int msg_flags;  
};
#define AF_UNSPEC       0  
define AF_UNIX 1  
define AF_INET6 10  
define AF_INET 2
#define PF_INET   AF_INET
#define PF_INET6  AF_INET6
#define PF_UNIX   AF_UNIX
#define PF_UNSPEC AF_UNSPEC
#define SOCK_STREAM 1
#define SOCK_PACKET 10
#define SOCK_DGRAM  2
#define SOCK_RAW   3
#define SOCK_RDM   4
#define SOCK_SEQPACKET 5
#define SO_DEBUG   1
#define SO_OOBINLINE 10
#define SO_NO_CHECK 11
#define SO_PRIORITY 12
#define SO_LINGER  13
#define SO_BSDCOMPAT 14
#define SO_REUSEADDR 2
#define SO_TYPE  3
#define SO_ACCEPTCONN 30
#define SO_ERROR   4
#define SO_DONTROUTE 5
#define SO_BROADCAST 6
#define SO_SNDBUF  7
#define SO_RCVBUF  8
#define SO_KEEPALIVE 9
#define SIOCGIFNAME 0x8910
#define SIOCGIFCONF 0x8912
#define SIOCGIFFLAGS 0x8913
#define SIOCGIFADDR 0x8915
#define SIOCGIFDSTADDR 0x8917
#define SIOCGIFBRDADDR 0x8919
#define SIOCGIFNETMASK 0x891b
#define SIOCGIFMTU 0x8921
#define SIOCGIFHWADDR 0x8927
#define SHUT_RD  0
#define SHUT_WR  1
#define SHUT_RDWR 2
#define MSG_WAITALL 0x100
#define MSG_TRUNC  0x20
#define MSG_NOSIGNAL 0x4000
#define MSG_EOR   0x80
#define MSG_OOB   1
#define MSG_PEEK  2
#define MSG_DONTROUTE 4
#define MSG_CTRUNC 8

extern ssize_t __recv_chk(int, void *, size_t, size_t, int);
extern ssize_t __recvfrom_chk(int, void *, size_t, size_t, int,
struct sockaddr *, socklen_t *);
extern int accept(int __fd, struct sockaddr *__addr,
socklen_t * __addr_len);
extern int bind(int __fd, const struct sockaddr *__addr,
socklen_t __len);
extern int connect(int __fd, const struct sockaddr *__addr,
socklen_t __len);
extern int getnameinfo(const struct sockaddr *__sa, socklen_t __salen,
struct sockaddr **__local, socklen_t *__locallen, char * __host, socklen_t __hostlen, char * __serv, socklen_t __servlen, unsigned int __flags);
extern int getpeername(int __fd, struct sockaddr *__addr, socklen_t *__len);
extern int getsockopt(int __fd, int __level, int __optname, void * __optval, socklen_t *__optlen);
extern int listen(int __fd, int __n);
extern ssize_t recv(int __fd, void *__buf, size_t __n, int __flags);
extern ssize_t recvfrom(int __fd, void *__buf, size_t __n, int __flags, struct sockaddr *__addr, socklen_t *__addr_len);
extern ssize_t recvmsg(int __fd, struct msghdr *__message, int __flags);
extern ssize_t send(int __fd, const void *__buf, size_t __n, int __flags);
extern ssize_t sendmsg(int __fd, const struct msghdr *__message, int __flags);
extern ssize_t sendto(int __fd, const void *__buf, size_t __n, int __flags, const struct sockaddr *__addr, socklen_t __addr_len);
extern int setsockopt(int __fd, int __level, int __optname, const void *__optval, socklen_t __optlen);
extern int shutdown(int __fd, int __how);
extern int sockatmark(int __fd);
extern int socket(int __domain, int __type, int __protocol);
extern int socketpair(int __domain, int __type, int __protocol, int __fds[2]);

14.4.84 sys/stat.h

#define S_ISBLK (m) (((m)&S_IFMT)==S_IFBLK)
#define S_ISCHR (m) (((m)&S_IFMT)==S_IFCHR)
#define S_ISDIR (m) (((m)&S_IFMT)==S_IFDIR)
#define S_ISFIFO (m) (((m)&S_IFMT)==S_IFIFO)
#define S_ISLNK (m) (((m)&S_IFMT)==S_IFLNK)
#define S_ISREG (m) (((m)&S_IFMT)==S_IFREG)
#define S_ISSOCK (m) (((m)&S_IFMT)==S_IFSOCK)
#define UTIME_NOW ((1l<<30)-1l)
#define UTIME_OMIT ((1l<<30)-2l)
#define S_TYPEISMQ(buf) ((buf)->st_mode - (buf)->st_mode)
#define S_TYPEISSSEM(buf) ((buf)->st_mode - (buf)->st_mode)
#define S_TYPEISSHM(buf) ((buf)->st_mode - (buf)->st_mode)
#define S_IRWXU (S_IREAD|S_IWRITE|S_IEXEC)
#define S_IROTH (S_IRGRP>>3)
#define S_IRGRP (S_IRUSR>>3)
#define S_IWROTH (S_IWUSR>>3)
#define S_IWGRP (S_IWUSR>>3)
#define S_IWOTH (S_IWUSR>>3)
#define S_IXROTH (S_IXGRP>>3)
#define S_IXGRP (S_IXUSR>>3)
#define S_IXOTH (S_IXUSR>>3)
#define S_ISVTX 01000
#define S_ISVTX 01000
#define S_IRUSR 0x0040
#define S_IWUSR 0x0080
#define S_IRUSR 0x0100
#define S_IWUSR 0x0200
#define S_ISGID 0x0400
#define S_ISGID 0x0800
#define S_ISGID 0x0400
#define S_ISGID 0x0800
#define S_IXUSR 0x1000
#define S_IXUSR 0x2000
#define S_IXUSR 0x4000
#define S_IXUSR 0x8000
#define S_IFIFO 0x1000
#define S_IFCHR 0x2000
#define S_IFDIR 0x4000
#define S_IFBLK 0x6000
#define S_IFREG 0x8000
#define S_IFLNK 0xa000
#define S_IFSOCK 0xc000
#define S_IFMT 0xf000
#define st_atime st_atim.tv_sec
#define st_ctime st_ctim.tv_sec
#define st_mtime st_mtim.tv_sec
#define S_IREAD S_IRUSR
#define S_IWRITE S_IWUSR
#define S_IEXEC S_IXUSR

extern int __fxstat(int __ver, int __fildes, struct stat *__stat_buf);
extern int __fxstat64(int __ver, int __fildes, struct stat64 *__stat_buf);
extern int __fxstatat(int __ver, int __fildes, const char *__filename,
                     struct stat *__stat_buf, int __flag);
extern int __fxstatat64(int __ver, int __fildes, const char *__filename,
                        struct stat64 *__stat_buf, int __flag);
extern int __lxstat(int __ver, const char *__filename, struct stat *__stat_buf);
extern int __lxstat64(int __ver, const char *__filename, struct stat64 *__stat_buf);
extern int __lxstat64(int __ver, const char *__filename, struct stat64 *__stat_buf);
extern int __lxstat4(int __ver, const char *__filename, struct stat4 *__stat_buf);
extern int __lxstat64(int __ver, const char *__filename, struct stat64 *__stat_buf);
extern int __lxstat64(int __ver, const char *__filename, struct stat64 *__stat_buf);
extern int chmod(const char *__file, mode_t __mode);
extern int fchmod(int __fd, mode_t __mode);
extern int fchmodat(int __fd, const char *__file, mode_t __mode, int __flag);
extern int futimens(int fd, const struct timespec times[2]);
extern int lstat(const char *__file, struct stat *__buf);
extern int lstat64(const char *__file, struct stat64 *__buf);
extern int mkdir(const char *__path, mode_t __mode);
extern int mkdirat(int __fd, const char *__path, mode_t __mode);
extern int mkfifo(const char *__path, mode_t __mode);
extern int mkfifoat(int __fd, const char *__path, mode_t __mode);
extern int mknod(const char *__path, mode_t __mode, dev_t __dev);
extern int mknodat(int __fd, const char *__path, mode_t __mode, dev_t __dev);
extern int chmod(int __fd, const char *__file, mode_t __mode, int __flag);
extern int lstat64(const char *__file, struct stat64 *__buf);
extern int mknod(const char *__path, mode_t __mode, dev_t __dev);
extern int utimensat(int fd, const char *path, const struct timespec times[2], int flags);

14.4.85 sys/statfs.h

#define NFS_SUPER_MAGIC 0x6969
14.4.86 sys/statvfs.h

extern int fstatvfs(int __fildes, struct statvfs *__buf);
extern int fstatvfs64(int __fildes, struct statvfs64 *__buf);
extern int statvfs(const char *__file, struct statvfs *__buf);
extern int statvfs64(const char *__file, struct statvfs64 *__buf);

14.4.87 sys/sysinfo.h

struct sysinfo {
    long int uptime;            /* Seconds since boot */
    unsigned long int loads[3]; /* 1, 5, and 15 minute load averages */
    unsigned long int totalram; /* Total usable main memory size */
    unsigned long int freeram; /* Available memory size */
    unsigned long int sharedram; /* Amount of shared memory */
    unsigned long int bufferram; /* Memory used by buffers */
    unsigned long int totalswap; /* Total swap space size */
    unsigned long int freeswap; /* Swap space still available */
    unsigned short procs;       /* Number of current processes */
    unsigned short pad;         /* Padding for m68k */
    unsigned long int totalhigh; /* Total high memory size */
    unsigned long int freehigh; /* Available high memory size */
    unsigned int mem_unit;      /* Memory unit size in bytes */
    char _f[20 - 2 * sizeof(long) - sizeof(int)]; /* Padding for libc5 */
};
extern int sysinfo(struct sysinfo *info);

14.4.88 sys/time.h

#define ITIMER_REAL     0
#define ITIMER_VIRTUAL  1
#define ITIMER_PROF     2

struct timezone {
    int tz_minuteswest;
    int tz_dsttime;
};
typedef int __itimer_which_t;

struct itimerval {
    struct timeval it_interval;
    struct timeval it_value;
};
extern int adjtime(const struct timeval *__delta,
                    struct timeval *__olddelta);
extern int futimes(int fd, const struct timeval tv[2]);
extern int getitimer(__itimer_which_t __which, struct itimerval *__value);
extern int gettimeofday(struct timeval *__tv, struct timezone *__tz);
extern int lutimes(const char *filename, const struct timeval tv[2]);
extern int setitimer(__itimer_which_t __which, const struct itimerval *__new, struct itimerval *__old);
extern int utimes(const char *__file, const struct timeval *__tvp);

14.4.89 sys/timeb.h

struct timeb {
    time_t time;                /* Seconds since epoch, as from time. */
    unsigned short millitm;     /* Additional milliseconds. */
    short timezone;             /* Minutes west of GMT. */
    short dstflag;              /* Nonzero if Daylight Savings Time used. */
};
extern int ftime(struct timeb *__timebuf);

14.4.90 sys/times.h

struct tms {
    clock_t tms_utime;
    clock_t tms_stime;
    clock_t tms_cutime;
    clock_t tms_cstime;
};
extern clock_t times(struct tms *__buffer);

14.4.91 sys/types.h

#ifndef FALSE
#define FALSE   0
#endif
#ifndef TRUE
#define TRUE    1
#endif

typedef unsigned char u_int8_t;
typedef unsigned short u_int16_t;
typedef unsigned int u_int32_t;
typedef unsigned long long int u_int64_t;
typedef unsigned int uid_t;
typedef int pid_t;
typedef long int off_t;
typedef long long int off64_t;
typedef int key_t;
typedef unsigned int u_int;
typedef struct {
    int __val[2];
} fsid_t;
typedef unsigned int useconds_t;
typedef long int blksize_t;
typedef long int fd_mask;
typedef void *timer_t;
typedef int clockid_t;

typedef unsigned int id_t;
typedef unsigned long long int ino64_t;
typedef long long int loff_t;
typedef long int blkcnt_t;
typedef unsigned long int fsblkcnt_t;
typedef unsigned long int fsfilcnt_t;
typedef long long int blkcnt64_t;
typedef unsigned long long int fsblkcnt64_t;
typedef unsigned long long int fsfilcnt64_t;
typedef unsigned char u_char;
typedef unsigned short u_short;
typedef unsigned long int u_long;
typedef unsigned long int ino_t;
typedef unsigned int gid_t;
typedef unsigned long long int dev_t;
typedef unsigned int mode_t;
typedef unsigned long int nlink_t;
typedef char *caddr_t;
typedef long int clock_t;
typedef long int time_t;

14.4.92 sys/uio.h

extern ssize_t readv(int __fd, const struct iovec *__iovec, int __count);
extern ssize_t writev(int __fd, const struct iovec *__iovec, int __count);

14.4.93 sys/un.h

struct sockaddr_un {
    sa_family_t sun_family;     /* AF_UNIX */
    char sun_path[108];
};

14.4.94 sys/utsname.h

#define SYS_NMLN 65

struct utsname {
    char sysname[65];
    char nodename[65];
    char release[65];
    char version[65];
    char machine[65];
    char domainname[65];
};
extern int uname(struct utsname *__name);

14.4.95 sys/wait.h

#define WIFSIGNALED(status)   (!WIFSTOPPED(status) && !
                          WEXITED(status))
#define WIFSTOPPED(status)    (((status) & 0xff) == 0x7f)
#define WEXITED(status)       (((status) & 0xff00) >> 8)
#define WTERMSIG(status)      ((status) & 0x7f)
#define WCOREDUMP(status)     ((status) & 0x80)
#define WIFEXITED(status)     (WTERMSIG(status) == 0)
#define WNOHANG 0x00000001
#define WUNTRACED 0x00000002
#define WCOREFLAG 0x80
#define WSTOPSIG(status) WEXITSTATUS(status)

typedef enum {
    P_ALL,
    P_PID,
    P_PGID
} idtype_t;

extern pid_t wait(int *__stat_loc);
extern pid_t wait4(pid_t __pid, int *__stat_loc, int __options,
                    struct rusage *__usage);
extern int waitid(idtype_t __idtype, id_t __id, siginfo_t *__infop,
                   int __options);
extern pid_t waitpid(pid_t __pid, int *__stat_loc, int __options);

14.4.96 sysexits.h

#define EX_OK 0 /* successful termination */
#define EX_USAGE 64 /* command line usage error */
#define EX_BASE 64 /* base value for error messages */
#define EX_DATAERR 65 /* data format error */
#define EX_NOINPUT 66 /* cannot open input */
#define EX_NOUSER 67 /* addressse unknown */
#define EX_NOHOST 68 /* host name unknown */
#define EX_UNAVAILABLE 69 /* service unavailable */
#define EXSOFTWARE 70 /* internal software error */
#define EX_OSERR 71 /* system error (e.g., cannot fork) */
#define EX_OSFIL 72 /* critical OS file missing */
#define EX_CANTCREATE 73 /* cannot create (user) output file */
#define EX_IOERR 74 /* input/output error */
#define EX_TEMPFAIL 75 /* temp failure; user is invited to retry */
#define EX_PROTOCOL 76 /* remote error in protocol */
#define EX_NOPERM 77 /* permission denied */
#define EX_CONFIG 78 /* configuration error */

14.4.97 syslog.h

#define LOG_MAKEPRI(fac, pri) (((fac) << 3) | (pri))
#define LOG_PRI(p) ((p) & LOG_PRIMASK) /* extract priority */
#define LOG_EMERG 0 /* system is unusable */
#define LOG_PRIMASK 0x07 /* mask to extract priority part */
#define LOG_ALERT 1 /* action must be taken immediately */
#define LOG_CRIT 2 /* critical conditions */
#define LOG_ERR 3 /* error conditions */
#define LOG_WARNING 4 /* warning conditions */
#define LOG_NOTICE 5 /* normal but significant condition */
#define LOG_INFO 6 /* informational */
#define LOG_DEBUG 7 /* debug-level messages */

#define LOG_FAC(p) (((p) & LOG_FACMASK) >> 3) /* facility of pri */
#define LOG_KERN  (0<<3) /* kernel messages */
#define LOG_AUTHPRIV (10<<3) /* security/authorization messages (private) */
#define LOG_FTP (11<<3) /* ftp daemon */
#define LOG_USER (1<<3) /* random user-level messages */
#define LOG_MAIL (2<<3) /* mail system */
#define LOG_DAEMON (3<<3) /* system daemons */
#define LOG_AUTH (4<<3) /* security/authorization messages */
#define LOG_SYSLOG (5<<3) /* messages generated internally by syslogd */
#define LOG_LPR (6<<3) /* line printer subsystem */
#define LOG_NEWS (7<<3) /* network news subsystem */
#define LOG_UUCP (8<<3) /* UUCP subsystem */
#define LOG_CRON (9<<3) /* clock daemon */
#define LOG_FACMASK 0x03f8 /* mask to extract facility part */
#define LOG_LOCAL0 (16<<3) /* reserved for local use */
#define LOG_LOCAL1 (17<<3) /* reserved for local use */
#define LOG_LOCAL2 (18<<3) /* reserved for local use */
#define LOG_LOCAL3 (19<<3) /* reserved for local use */
#define LOG_LOCAL4 (20<<3) /* reserved for local use */
#define LOG_LOCAL5 (21<<3) /* reserved for local use */
#define LOG_LOCAL6 (22<<3) /* reserved for local use */
#define LOG_LOCAL7 (23<<3) /* reserved for local use */
#define LOG_UPTO(pri)  ((1 << ((pri)+1)) - 1) /* all priorities through pri */
#define LOG_MASK(pri)  (1 << (pri)) /* mask for one priority */
#define LOG_PID 0x01 /* log the pid with each message */
#define LOG_CONS 0x02 /* log on the console if errors in sending */
#define LOG_ODELAY 0x04 /* delay open until first syslog() (default) */
#define LOG_NDELAY 0x08 /* don't delay open */
#define LOG_NOWAIT 0x10 /* don't wait for console forks: DEPRECATED */
#define LOG_PERROR 0x20 /* log to stderr as well */

extern void __syslog_chk(int, int, const char *, ...);
extern void __vsyslog_chk(int, int, const char *, va_list);
extern void closelog(void);
extern int setlogmask(int __mask);
extern void syslog(int __pri, const char *__fmt, ...);
extern void vsyslog(int __pri, const char *__fmt, va_list __ap);

14.4.98 tar.h

#define REGTYPE '0'
define LNKTYPE '1'
define SYMTYPE '2'
define CHRTYPE '3'
define BLKTYPE '4'
define DRTYPE '5'
define FIFOTYPE '6'
define CONTTYPE '7'
define AREGTYPE '\0'
define TVERSION "00"
define TOEXEC 00001
```c
#define TOWRITE 00002
#define TOREAD 00004
#define TGEXEC 00010
#define TGWRITE 00020
#define TGREAD 00040
#define TUEXEC 00100
#define TUWRITE 00200
#define TUREAD 00400
#define TVERSLEN 2
#define TMAGLEN 6
#define TMAGIC "ustar"

14.4.99 termios.h

#define TCIFLUSH 0
#define TCOFF 0
#define TCSANOW 0
#define BS0 0000000
#define CR0 0000000
#define FF0 0000000
#define NL0 0000000
#define TAB0 0000000
#define VT0 0000000
#define OPOST 0000001
#define OCRNL 0000010
#define ONOCR 0000020
#define ONLRET 0000040
#define OFILL 0000100
#define OFDEL 0000200
#define NL1 0000400
#define TCOFLUSH 1
#define TCOON 1
#define TCSADRAIN 1
#define TCIOFF 2
#define TCI0FLUSH 2
#define TCSAFLUSH 2
#define TCION 3

typedef unsigned int speed_t;
typedef unsigned char cc_t;
typedef unsigned int tcflag_t;

#define NCCS 32

struct termios {
    tcflag_t c_iflag;    /* input mode flags */
    tcflag_t c_oflag;    /* output mode flags */
    tcflag_t c_cflag;    /* control mode flags */
    tcflag_t c_lflag;    /* local mode flags */
    cc_t c_line;        /* line discipline */
    cc_t c_cc[NCCS];    /* control characters */
    speed_t c_ispeed;   /* input speed */
    speed_t c_ospeed;   /* output speed */
};

#define VINTR 0
#define VQUIT 1
#define VLNEXT 15
#define VERASE 2
#define VKILL 3
#define VEOF 4
```
14.4.100 time.h

#define CLK_TCK ((clock_t)sysconf(2))
#define timermax((tvp)->tv_sec = (tvp)->tv_usec = 0)
#define timerisset((tvp)->tv_sec || (tvp)->tv_usec)
#define CLOCK_REALTIME 0
#define CLOCK_MONOTONIC 1
#define TIMER_ABSTIME 1
#define CLOCKS_PER_SEC 10000001
#define CLOCK_PROCESS_CPUTIME_ID 2
#define CLOCK_THREAD_CPUTIME_ID 3
#define timeradd(a,b,result) \ do { \ (result)->tv_sec = (a)->tv_sec + (b)->tv_usec; \}
(result)->tv_usec = (a)->tv_usec + (b)->tv_usec; \n if (((result)->tv_usec >= 1000000) \n \n ++(result)->tv_sec; \n (result)->tv_usec -= 1000000; \n } \n } while (0)
#define timersub(a,b,result)    \n do { \n (result)->tv_sec = (a)->tv_sec - (b)->tv_sec; \n (result)->tv_usec = (a)->tv_usec - (b)->tv_usec; \n if ((result)->tv_usec < 0) { \n --(result)->tv_sec; \n (result)->tv_usec += 1000000; \n } \n } while (0)
#define timercmp(a,b,CMP)       \n (((a)->tv_sec == (b)->tv_sec) ? \n ((a)->tv_usec CMP (b)->tv_usec) : \n (a)->tv_sec CMP (b)->tv_usec))

struct tm {
   int tm_sec;
   int tm_min;
   int tm_hour;
   int tm_mday;
   int tm_mon;
   int tm_year;
   int tm_wday;
   int tm_yday;
   int tm_isdst;
   long int tm_gmtoff;
   char *tm_zone;
};

struct timespec {
   time_t tv_sec;
   long int tv_nsec;
};

struct itimerspec {
   struct timespec it_interval;
   struct timespec it_value;
};

extern int __daylight;
extern long int __timezone;
extern char *__tzname[];
extern char *asctime(const struct tm *__tp);
extern char *asctime_r(const struct tm *__tp, char *__buf);
extern clock_t clock(void);
extern  int  clock_getcpuclockid(pid_t  __pid,  clockid_t  *__clock_id);
extern  int clock_getres(clockid_t  __clock_id, struct  timespec *__res);
extern int clock_gettime(clockid_t __clock_id, struct timespec *__tp);
extern int clock_nanosleep(clockid_t __clock_id, int __flags, const struct timespec *__req, struct timespec *__rem);
extern int clock_settime(clockid_t __clock_id, const struct timespec *__tp);
extern char *ctime(const time_t * __timer);
extern char *ctime_r(const time_t * __timer, char *__buf);
extern int daylight;
extern double difftime(time_t __time1, time_t __time0);
extern struct tm *getdate(const char *__string);
extern int getdate_err;
extern struct tm *gmtime(const time_t * __timer);
extern struct tm *gmtime_r(const time_t * __timer, struct tm *__tp);
extern struct tm *localtime(const time_t * __timer);
extern struct tm *localtime_r(const time_t * __timer, struct tm *__tp);
extern time_t mktime(struct tm *__tp);
extern int nanosleep(const struct timespec *__requested_time,
                     struct timespec *__remaining);
extern int stime(const time_t * __when);
extern size_t strftime(char *__s, size_t __maxsize, const char *__format,
                        const struct tm *__tp);
extern size_t strftime_l(char *s, size_t maxsize, const char *format,
                          const struct tm *__tp);
extern char *strptime(const char *__s, const char *__fmt, struct tm *__tp);
extern time_t time(time_t * __timer);
extern int timer_create(clockid_t __clock_id, struct sigevent *__evp,
                         timer_t *__timerid);
extern int timer_delete(timer_t __timerid);
extern int timer_getoverrun(timer_t __timerid);
extern int timer_gettime(timer_t __timerid, struct itimerspec *__value);
extern int timer_settime(timer_t __timerid, int __flags,
                          const struct itimerspec *__value,
                          struct itimerspec *__ovalue);
extern long int timezone;
extern char *tzname[];
extern void tzset(void);

14.4.101 ucontext.h

extern int getcontext(ucontext_t *__ucp);
extern void makecontext(ucontext_t *__ucp, void (*__func)(void),
                        int __argc, ...);
extern int setcontext(const struct ucontext *__ucp);
extern int swapcontext(ucontext_t *__oucp, const struct ucontext *__ucp);

14.4.102 ulimit.h

#define UL_GETFSIZE     1
#define UL_SETFSIZE     2

extern long int ulimit(int __cmd, ...);

14.4.103 unistd.h

#ifndef SEEK_SET
#define SEEK_SET 0
#endif
#define STDIN_FILENO 0
#ifndef SEEK_CUR
#define SEEK_CUR 1
#endif
#define STDOUT_FILENO 1
#ifndef SEEK_END
#define SEEK_END 2
#endif

#define STDERR_FILENO 2
#define F_OK 0
#define X_OK 1
#define W_OK 2
#define R_OK 4

#define _POSIX_VDISABLE '\0'
#define _POSIX_ASYNC_IO 0
#define _POSIX_CHOWN_RESTRICTED 1
#define _POSIX_JOB_CONTROL 1
#define _POSIX_NO_TRUNC 1
#define _POSIX2_VERSION 200809L
#define _POSIX2_C_BIND 200809L
#define _POSIX2_LOCALEDEF 200809L
#define _POSIX2_VERSION 200809L
#define _POSIX_ADVISORY_INFO 200809L
#define _POSIX_BARRIERS 200809L
#define _POSIX_CLOCK_SELECTION 200809L
#define _POSIX_FSYNC 200809L
#define _POSIX_IPV6 200809L
#define _POSIX_MAPPED_FILES 200809L
#define _POSIX_MEMLOCK 200809L
#define _POSIX_MEMLOCK_RANGE 200809L
#define _POSIX_MEMORY_PROTECTION 200809L
#define _POSIX_Message Passing 200809L
#define _POSIX_PRIORITIZED_IO 200809L
#define _POSIX_PRIORITY_SCHEDULING 200809L
#define _POSIX_RAW_SOCKETS 200809L
#define _POSIX_READER_WRITER_LOCKS 200809L
#define _POSIX_REALTIME_SIGNALS 200809L
#define _POSIX_SEMAPHORES 200809L
#define _POSIX_SHARED_MEMORY_OBJECTS 200809L
#define _POSIX_SPAWN 200809L
#define _POSIX_SPIN_LOCKS 200809L
#define _POSIX_SYNCHRONIZED_IO 200809L
#define _POSIX_THREADS 200809L
#define _POSIX_THREAD_ATTR_STACKADDR 200809L
#define _POSIX_THREAD_ATTR_STACKSIZE 200809L
#define _POSIX_THREAD_PRIORITY_SCHEDULING 200809L
#define _POSIX_THREAD_PRIO_INHERIT 200809L
#define _POSIX_THREAD_PRIO_PROTECT 200809L
#define _POSIX_THREAD_PROCESS_SHARED 200809L
#define _POSIX_THREAD_SAFE_FUNCTIONS 200809L
#define _POSIX_TIMEOUTS 200809L
#define _POSIX_TIMES 200809L
#define _POSIX_VERSION 200809L

#define _PC_LINK_MAX 0
#define _PC_MAX_CANON 1
#define _PC_ASYNC_IO 10
#define _PC_PRIO_IO 11
#define _PC_FILESIZEBITS 13
#define _PC_REC_INCR_XFER_SIZE 14
#define _PC_REC_MIN_XFER_SIZE 16
#define _PC_REC_XFER_ALIGN 17
#define _PC_ALLOC_SIZE_MIN 18
#define _PC_MAX_INPUT 2
#define _PC_2_SYMLINKS 20
#define _PC_NAME_MAX 3
#define _PC_PATH_MAX 4
#define _PC_PIPE_BUF 5
#define _PC_CHOWN_RESTRICTED 6
#define _PC_NO_TRUNC 7
#define _PC_VDISABLE 8
#define _PC_SYNC_IO 9

#define _SC_ARG_MAX 0
#define _SC_CHILD_MAX 1
#define _SC_PRIORITY_SCHEDULING 10
#define _SC_XOPEN_XPG4 100
#define _SC_CHAR_BIT 101
#define _SC_CHAR_MAX 102
#define _SC_CHAR_MIN 103
#define _SC_INT_MAX 104
#define _SC_INT_MIN 105
#define _SC_LONG_BIT 106
#define _SC_WORD_BIT 107
#define _SC_MB_LEN_MAX 108
#define _SC_NZERO 109
#define _SC_TIMERS 11
#define _SC_SSIZE_MAX 110
#define _SC_SCHAR_MAX 111
#define _SC_SCHAR_MIN 112
#define _SC_SHRT_MAX 113
#define _SC_SHRT_MIN 114
#define _SC_UCHAR_MAX 115
#define _SC_UINT_MAX 116
#define _SC_USHRT_MAX 117
#define _SC_UCHAR_MAX 118
#define _SC_UCHAR_MAX 119
#define _SC_SYNCHRONOUS_IO 12
#define _SC_NL_ARGMAX 120
#define _SC_NL_LANGMAX 121
#define _SC_NL_MSGMAX 122
#define _SC_NL_NMAX 123
#define _SC_NL_SETMAX 124
#define _SC_NL_TEXTMAX 125
#define _SC_XBS5_ILP32_OFF32 126
#define _SC_XBS5_ILP32_OFFBIG 127
#define _SC_XBS5_LP64_OFF64 128
#define _SC_XBS5_LPBIG_OFFBIG 129
#define _SC_XOPEN_LEGACY 130
#define _SC_PRIORITIZED_IO 131
#define _SC_XOPEN_REALTIME_THREADS 132
#define _SC_ADVISORY_INFO 133
#define _SC_BARRIERS 134
#define _SC_BASE 135
#define _SC_CLOCK_SELECTION 136
#define _SC_CPUTIME 137
#define _SC_THREAD_CPUTIME 138
#define _SC_SYNCHRONIZED_IO 139
#define _SC_DEVICE_IE 140
#define _SC_DEVICE_SPECIFIC 141
#define _SC_DEVICE_SPECIFIC_R 142
#define _SC_FD_MGMT 143
#define _SC_FIFO 144
#define _SC_PIPE 145
#define _SC_FILE_ATTRIBUTES 146
#define _SC_FILE_LOCKING 147
#define _SC_FILE_SYSTEM 148
#define _SC_MONOTONIC_CLOCK 149
#define _SC_FSYNC 150
#define _SC_MULTI_PROCESS 151
#define _SC_SINGLE_PROCESS 152
#define _SC_NETWORKING 152
#define _SC_READER_WRITER_LOCKS 153
#define _SC_SPIN_LOCKS 154
#define _SC_REGEXP 155
#define _SC_REGEX_VERSION 156
#define _SC_SHELL 157
#define _SC_SIGNALS 158
#define _SC_SPAWN 159
#define _SC_MAPPED_FILES 16
#define _SC_SPORADIC_SERVER 160
#define _SC_THREAD_SPORADIC_SERVER 161
#define _SC_SYSTEM_DATABASE 162
#define _SC_SYSTEM_DATABASE_R 163
#define _SC_TIMEOUTS 164
#define _SC_TYPED_MEMORY_OBJECTS 165
#define _SC_USER_GROUPS 166
#define _SC_USER_GROUPS_R 167
#define _SC_2_PBS 168
#define _SC_2_PBS_ACCOUNTING 169
#define _SC_MEMLOCK 17
#define _SC_2_PBSLOCATE 170
#define _SC_2_PBSMESSAGE 171
#define _SC_2_PBSTRACK 172
#define _SC_SYMLOOP_MAX 173
#define _SC_STREAMS 174
#define _SC_2_PBSCHECKPOINT 175
#define _SC_V6_ILP32_OFF32 176
#define _SC_V6_ILP32_OFFBIG 177
#define _SC_V6_LP64_OFF64 178
#define _SC_V6_LPBIG_OFFBIG 179
#define _SC_MEMLOCK_RANGE 180
#define _SC_HOST_NAME_MAX 181
#define _SC_TRACE 182
#define _SC_TRACE_EVENT_FILTER 183
#define _SC_LEVEL1_ICACHE_SIZE 184
#define _SC_LEVEL1_ICACHE_ASSOC 185
#define _SC_LEVEL1_ICACHE_LINESIZE 186
#define _SC_LEVEL1_DCACHE_SIZE 187
#define _SC_LEVEL1_DCACHE_ASSOC 188
#define _SC_MEMORY_PROTECTION 189
#define _SC_LEVEL1_DCACHE_LINESIZE 190
#define _SC_LEVEL2_CACHE_SIZE 191
#define _SC_LEVEL2_CACHE_ASSOC 192
#define _SC_LEVEL2_CACHE_LINESIZE 193
#define _SC_LEVEL3_CACHE_SIZE 194
#define _SC_LEVEL3_CACHE_ASSOC 195
#define _SC_LEVEL3_CACHE_LINESIZE 196
#define _SC_LEVEL4_CACHE_SIZE 197
#define _SC_LEVEL4_CACHE_ASSOC 198
#define _SC_LEVEL4_CACHE_LINESIZE 199
#define _SC_CLK_TCK 2
#define _SC_MESSAGE_PASSING 20
#define _SC_SEMAPHORES 21
#define _SC_AIO_LISTIO_MAX 22
#define _SC_IPV6 23
#define _SC_RAW_SOCKETS 235
#define _SC_V7_ILP32_OFF32 236
#define _SC_V7_ILP32_OFFBIG 237
#define _SC_V7_LP64_OFF64 238
#define _SC_V7_LPBIG_OFFBIG 239
#define _SC_AIO_MAX 24
#define _SC_SS_REPL_MAX 240
#define _SC_TRACE_EVENT_NAME_MAX 242
#define _SC_TRACE_NAME_MAX 243
#define _SC_TRACE_SYS_MAX 244
#define _SC_TRACE_USER_EVENT_MAX 245
#define _SC_XOPEN_STREAMS 246
#define _SC_THREAD_ROBUST_PRIO_INHERIT 247
#define _SC_THREAD_ROBUST_PRIO_PROTECT 248
#define _SC_AIO_PRIO_DELTA_MAX 25
#define _SCDELAYTIMER_MAX 26
#define _SC_MQ_OPEN_MAX 27
#define _SC_MQ_PRIO_MAX 28
#define _SC_VERSION 29
#define _SC_NGROUPS_MAX 3
#define _SC_PAGESIZE 30
#define _SC_PAGE_SIZE 30
#define _SC_RTSIG_MAX 31
#define _SC_SEM_NSEMS_MAX 32
#define _SC_SEM_VALUE_MAX 33
#define _SC_SIGQUEUE_MAX 34
#define _SC_TIMER_MAX 35
#define _SC_BC_BASE_MAX 36
#define _SC_BC_DIM_MAX 37
#define _SC_BC_SCALE_MAX 38
#define _SC_BC_STRING_MAX 39
#define _SC_Open_MAX 4
#define _SC_COLL_WEIGHTS_MAX 40
#define _SC_EQUIV_CLASS_MAX 41
#define _SC_EXPR_NEST_MAX 42
#define _SC_LINE_MAX 43
#define _SC_RE_DUP_MAX 44
#define _SC_CHARCLASS_NAME_MAX 45
#define _SC_2_VERSION 46
#define _SC_2_C_BIND 47
#define _SC_2_C_DEV 48
#define _SC_2_FORT_DEV 49
#define _SC_2_FORT_RUN 50
#define _SC_2_SW_DEV 51
#define _SC_2_LOCALEDEF 52
#define _SC_PII 53
#define _SC_PII_XTI 54
#define _SC_PII_SOCKET 55
#define _SC_PII_INTERNET 56
#define _SC_PII_OSI 57
#define _SC_POLL 58
#define _SC_SELECT 59
#define _SC_TZNAME_MAX 6
#define _SC_JIOV_MAX 60
#define _SC_UIO_MAXIOV 60
#define _SC_PII_INTERNET_STREAM 61
#define _SC_PII_INTERNET_DGRAM 62
#define _SC_PII_OSI_COTS 63
#define _SC_PII_OSI_CLTS 64
#define _SC_PII_OSI_M 65
#define _SC_IIOV_MAX 66
#define _SC_THREADS 67
#define _SC_THREAD_SAFE_FUNCTIONS 68
#define _SC_GETGR_R_SIZE_MAX 69
#define _SC_JOB_CONTROL 7
#define _SC_GETPW_R_SIZE_MAX 70
#define _SC_LOGIN_NAME_MAX 71
#define _SC_TTY_NAME_MAX 72
#define _SC_THREAD_DESTRUCTOR_ITERATIONS 73
#define _SC_THREAD_KEYS_MAX 74
#define _SC_THREAD_STACK_MIN 75
#define _SC_THREAD_THREADS_MAX 76
#define _SC_THREAD_ATTR_STACKADDR 77
#define _SC_THREAD_ATTR_STACKSIZE       78
#define _SC_THREAD_PRIORITY_SCHEDULING  79
#define _SC_SAVED_IDS   8
#define _SC_THREAD_PRIO_INHERIT 80
#define _SC_THREAD_PRIO_PROTECT 81
#define _SC_THREAD_PROCESS_SHARED       82
#define _SC_NPROCESSORS_CONF    83
#define _SC_NPROCESSORS_ONLN 84
#define _SC_PHYS_PAGES 85
#define _SC_AVPHYS_PAGES 86
#define _SC_ATEXIT_MAX 87
#define _SC_PASS_MAX 88
#define _SC_XOPEN_VERSION 89
#define _SC_REALTIME_SIGNALS 9
#define _SC_XOPEN_XCU_VERSION 90
#define _SC_XOPEN_UNIX 91
#define _SC_XOPEN_CRYPT 92
#define _SC_XOPEN_ENH_I18N 93
#define _SC_XOPEN_SHM 94
#define _SC_2_CHAR_TERM 95
#define _SC_2_C_VERSION 96
#define _SC_2_UPE 97
#define _SC_XOPEN_XPG2 98
#define _SC_XOPEN_XPG3 99
#define _CS_PATH 0
#define _POSIX_REGEXP 1
#define _CS_XBS5_ILP32_OFF32_CFLAGS 1100
#define _CS_XBS5_ILP32_OFF32_LDFLAGS 1101
#define _CS_XBS5_ILP32_OFF32_LIBS 1102
#define _CS_XBS5_ILP32_OFF32_LINTFLAGS 1103
#define _CS_XBS5_ILP32_OFFBIG_CFLAGS 1104
#define _CS_XBS5_ILP32_OFFBIG_LDFLAGS 1105
#define _CS_XBS5_ILP32_OFFBIG_LIBS 1106
#define _CS_XBS5_ILP32_OFFBIG_LINTFLAGS 1107
#define _CS_XBS5_LP64_OFF64_CFLAGS 1108
#define _CS_XBS5_LP64_OFF64_LDFLAGS 1109
#define _CS_XBS5_LP64_OFF64_LIBS 1110
#define _CS_XBS5_LP64_OFF64_LINTFLAGS 1111
#define _CS_XBS5_LPBIG_OFFBIG_CFLAGS 1112
#define _CS_XBS5_LPBIG_OFFBIG_LDFLAGS 1113
#define _CS_XBS5_LPBIG_OFFBIG_LIBS 1114
#define _CS_XBS5_LPBIG_OFFBIG_LINTFLAGS 1115
#define _CS_POSIX_V6_ILP32_OFF32_CFLAGS 1116
#define _CS_POSIX_V6_ILP32_OFF32_LDFLAGS 1117
#define _CS_POSIX_V6_ILP32_OFF32_LIBS 1118
#define _CS_POSIX_V6_ILP32_OFF32_LINTFLAGS 1119
#define _CS_POSIX_V6_ILP32_OFFBIG_CFLAGS 1120
#define _CS_POSIX_V6_ILP32_OFFBIG_LDFLAGS 1121
#define _CS_POSIX_V6_ILP32_OFFBIG_LIBS 1122
#define _CS_POSIX_V6_ILP32_OFFBIG_LINTFLAGS 1123
#define _CS_POSIX_V6_LP64_OFF64_CFLAGS 1124
#define _CS_POSIX_V6_LP64_OFF64_LDFLAGS 1125
#define _CS_POSIX_V6_LP64_OFF64_LIBS 1126
#define _CS_POSIX_V6_LP64_OFF64_LINTFLAGS 1127
#define _CS_POSIX_V6_LPBIG_OFFBIG_CFLAGS 1128
#define _CS_POSIX_V6_LPBIG_OFFBIG_LDFLAGS 1129
#define _CS_POSIX_V6_LPBIG_OFFBIG_LIBS 1130
#define _CS_POSIX_V6_LPBIG_OFFBIG_LINTFLAGS 1131
#define _CS_POSIX_V7_ILP32_OFF32_CFLAGS 1132
#define _CS_POSIX_V7_ILP32_OFF32_LDFLAGS 1133
#define _CS_POSIX_V7_ILP32_OFF32_LIBS 1134
#define _CS_POSIX_V7_ILP32_OFF32_LINTFLAGS 1135
#define _CS_POSIX_V7_ILP32_OFFBIG_CFLAGS 1136
#define _CS_POSIX_V7_ILP32_OFFBIG_LDFLAGS 1137
#define _CS_POSIX_V7_ILP32_OFFBIG_LIBS 1138
#define _CS_POSIX_V7_ILP32_OFFBIG_LINTFLAGS 1139
#define _CS_POSIX_V7_LP64_OFF64_CFLAGS 1140
#define _CS_POSIX_V7_LP64_OFF64_LDFLAGS 1141
#define _CS_POSIX_V7_LP64_OFF64_LIBS 1142
#define _CS_POSIX_V7_LP64_OFF64_LINTFLAGS 1143
#define _CS_POSIX_V7_LPBIG_OFFBIG_CFLAGS 1144
#define _CS_POSIX_V7_LPBIG_OFFBIG_LDFLAGS 1145
#define _CS_POSIX_V7_LPBIG_OFFBIG_LIBS 1146
#define _CS_POSIX_V7_LPBIG_OFFBIG_LINTFLAGS 1147
#define _CS_V6_ENV 1148
#define _CS_V7_ENV 1149

#define _XOPEN_XPG4 1
#define _XOPEN_VERSION 700

#define F_ULOCK 0
#define F_LOCK 1
#define F_TLOCK 2
#define F_TEST 3

extern size_t __confstr_chk(int, char *, size_t, size_t);
extern char **__environ;
extern char *__getcwd_chk(char *, size_t, size_t);
extern int __getgroups_chk(int, gid_t *, size_t);
extern int __gethostname_chk(char *, size_t, size_t);
extern int __getlogin_r_chk(char *, size_t, size_t);
extern pid_t __getpgid(pid_t __pid);
extern ssize_t __pread64_chk(int, void *, size_t, off64_t, size_t);
extern ssize_t __preadchk(int, void *, size_t, off_t, size_t);
extern ssize_t __read_chk(int, void *, size_t, size_t);
extern ssize_t __readlink_chk(const char *, char *, size_t, size_t);
extern int __ttyname_r_chk(int, char *, size_t, size_t);
extern char **_environ;
extern void _exit(int __status);
extern int access(const char *__name, int __type);
extern int acct(const char *__name);
extern unsigned int alarm(unsigned int __seconds);
extern int brk(void *__addr);
extern int chdir(const char *__path);
extern int chown(const char *__file, uid_t __owner, gid_t __group);
extern int chroot(const char *__path);
extern int close(int __fd);
extern size_t confstr(int __name, char *__buf, size_t __len);
extern char *crypt(const char *__key, const char *__salt);
extern char *ctermid(char *__s);
extern char *cuserid(char *__s);
extern int daemon(int __nochdir, int __noclose);
extern int dup(int __fd);
extern int dup2(int __fd, int __fd2);
extern void encrypt(char *__block, int __edflag);
extern int exec1(const char *__path, const char *__arg, ...);
extern int execle(const char *__path, const char *__arg, ...);
extern int execp(const char *__file, const char *__arg, ...);
extern int execv(const char *__path, char *__argv[]);
extern int execve(const char *__path, char *__argv[],
    char *const __envp[]);
extern int execvp(const char *__file, char *__argv[]);
extern int faccessat(int __fd, const char *__file, int __type,
    int __flag);
extern int fchdir(int __fd);
extern int fchown(int __fd, uid_t __owner, gid_t __group);
extern int fchownat(int __fd, const char *__file, uid_t __owner,
    gid_t __group, int __flag);
extern int fdatasync(int __fildes);
extern int fexecve(int __fd, char *const __argv[], char *const __envp[]);
extern pid_t fork(void);
extern long int fpathconf(int __fd, int __name);
extern int fsync(int __fd);
extern int ftruncate(int __fd, off_t __length);
extern int ftruncate64(int __fd, off64_t __length);
extern char *getcwd(char *__buf, size_t __len);
extern int getdomainname(char *__name, size_t __len);
extern int getdtablesize(void);
extern gid_t getegid(void);
extern uid_t geteuid(void);
extern gid_t getgid(void);
extern int getgroups(int __size, gid_t __list[]);
extern long int gethostid(void);
extern int gethostname(char *__name, size_t __len);
extern char *getlogin(void);
extern int getlogin_r(char *__name, size_t __name_len);
extern int getopt(int __argc, char *const __argv[], const char *__shortopts);
extern int getpagesize(void);
extern pid_t getpgid(pid_t __pid);
extern pid_t getpgrp(void);
extern pid_t getpid(void);
extern pid_t getppid(void);
extern pid_t getsid(pid_t __pid);
extern uid_t getuid(void);
extern char *getwd(char *__buf);
extern int isatty(int __fd);
extern long int lchown(const char *__file, uid_t __owner, gid_t __group);
extern long int lstat(const char *__path, struct.Statbuf __sb);
extern int link(const char *__from, const char *__to);
extern int linkat(int __fromfd, const char *__from, int __tofd, const char *__to, int __flags);
extern int lockf(int __fd, int __cmd, off_t __len);
extern int lockf64(int __fd, int __cmd, off64_t __len);
extern off_t lseek(int __fd, off_t __offset, int __whence);
extern loff_t lseek64(int __fd, loff_t __offset, int __whence);
extern int nice(int __inc);
extern char *optarg;
extern int opterr;
extern int optind;
extern int optopt;
extern long int pathconf(const char *__path, int __name);
extern int pause(void);
extern int pipe(int __pipedes[2]);
extern ssize_t pread(int __fd, void *__buf, size_t __nbytes, off_t __offset);
extern ssize_t pread64(int __fd, void *__buf, size_t __nbytes, off64_t __offset);
extern ssize_t pwrite(int __fd, const void *__buf, size_t __n, off_t __offset);
extern ssize_t pwrite64(int __fd, const void *__buf, size_t __n, off64_t __offset);
extern ssize_t read(int __fd, void *__buf, size_t __nbytes);
extern ssize_t readlink(const char *__path, char *__buf, size_t __len);
extern ssize_t readlinkat(int __fd, const char *__path, char *__buf, size_t __len);
extern int rmdir(const char *__path);
extern void *sbrk(intptr_t __delta);
extern int setegid(gid_t __gid);
extern int seteuid(uid_t __uid);
extern int setgid(gid_t __gid);
14 Base Libraries

extern int sethostname(const char *__name, size_t __len);
extern void setkey(const char *__key);
extern int setpgid(pid_t __pid, pid_t __pgid);
extern int setpgrp(void);
extern int setregid(gid_t __rgid, gid_t __egid);
extern int setreuid(uid_t __ruid, uid_t __euid);
extern pid_t setsid(void);
extern int setuid(uid_t __uid);
extern unsigned int sleep(unsigned int __seconds);
extern void swab(const void *__from, void *__to, ssize_t __n);
extern int symlink(const char *__from, const char *__to);
extern int symlinkat(const char *__from, int __tofd, const char *__to);
extern void sync(void);
extern long int sysconf(int __name);
extern pid_t tcgetpgrp(int __fd);
extern int tcsetpgrp(int __fd, pid_t __pgrp_id);
extern int truncate(const char *__file, off_t __length);
extern int truncate64(const char *__file, off64_t __length);
extern char *ttyname(int __fd);
extern int ttyname_r(int __fd, char *__buf, size_t __buflen);
extern unsigned int ualarm(useconds_t __value, useconds_t __interval);
extern int unlink(const char *__name);
extern int unlinkat(int __fd, const char *__name, int __flag);
extern int usleep(useconds_t __useconds);
extern pid_t vfork(void);
extern ssize_t write(int __fd, const void *__buf, size_t __n);

14.4.104 utime.h

struct utimbuf {
    time_t actime;
    time_t modtime;
};
extern int utime(const char *__file, const struct utimbuf *__file_times);

14.4.105 utmp.h

#define BASE_LIBRARIES 256
#define LINE_SIZE 32
#define NAME_SIZE 32
#define ut_addr ut_addr_v6[0]
#define ut_time ut_tv.tv_sec
#define ut_name ut_user         /* Backwards compatibility */

struct exit_status {
    short e_termination;        /* Process termination status. */
    short e_exit;               /* Process exit status. */
};

#define EMPTY 0               /* No valid user accounting
information. */
#define RUN_LVL 1               /* The system's runlevel. */
#define BOOT_TIME 2             /* Time of system boot. */
#define NEW_TIME 3              /* Time after system clock
changed. */
#define OLD_TIME 4              /* Time when system clock
changed. */
#define INIT_PROCESS 5          /* Process spawned by the init
process. */
#define LOGIN_PROCESS 6         /* Session leader of a logged in
#define USER_PROCESS  7       /* Normal process. */
#define DEAD_PROCESS   8       /* Terminated process. */
#define ACCOUNTING     9

extern void endutent(void);
extern struct utmp *getutent(void);
extern int getutent_r(struct utmp *__buffer, struct utmp *__result);
extern void login(const struct utmp *__entry);
extern int login_tty(int __fd);
extern void logout(const char *__ut_line);
extern void logwtmp(const char *__ut_line, const char *__ut_name, const char *__ut_host);
extern void setutent(void);
extern int utmpname(const char *__file);

14.4.106 utmpx.h

extern void endutxent(void);
extern struct utmpx *getutxent(void);
extern struct utmpx *getutxid(const struct utmpx *__id);
extern struct utmpx *getutxline(const struct utmpx *__line);
extern struct utmpx *pututxline(const struct utmpx *__utmpx);
extern void setutxent(void);

14.4.107 wchar.h

#define WEOF    (0xffffffffu)
#define WCHAR_MAX       0x7FFFFFFF
#define WCHAR_MIN       0x80000000
typedef unsigned long int wctype_t;
typedef const int32_t *wctrans_t;

extern wchar_t *__fgetws_chk(wchar_t *, size_t, int, FILE *);
extern wchar_t *__fgetws_unlocked_chk(wchar_t *, size_t, int, FILE *);
extern int __fwprintf_chk(FILE *, int, const wchar_t *, ...);
extern size_t __mbsnrtowcs_chk(wchar_t *, const char **, size_t, size_t, mbstate_t *, size_t);
extern size_t __mbsrtowcs_chk(wchar_t *, const char **, size_t, mbstate_t *, size_t);
extern int __swprintf_chk(wchar_t *, size_t, int, size_t, const wchar_t *, ...
extern int __vfwprintf_chk(FILE *, int, const wchar_t *, va_list);
extern int __vswprintf_chk(wchar_t *, size_t, int, size_t, const wchar_t *, va_list);
extern int __vwprintf_chk(int, const wchar_t *, va_list);
extern wchar_t *__wcpcpy_chk(wchar_t *, const wchar_t *, size_t);
extern wchar_t *__wcpncpy_chk(wchar_t *, const wchar_t *, size_t, size_t);
extern size_t __wcrtomb_chk(char *, wchar_t, mbstate_t *, size_t);
extern wchar_t *__wcscat_chk(wchar_t *, const wchar_t *, size_t);
extern wchar_t *__wcsncat_chk(wchar_t *, const wchar_t *, size_t, size_t);
extern wchar_t *__wcsncpy_chk(wchar_t *, const wchar_t *, size_t, size_t,
extern size_t __wcsnrtombs_chk(char *, const wchar_t * *, size_t, size_t, mbstate_t *, size_t);
extern size_t __wcsrtombs_chk(char *, const wchar_t * *, size_t, mbstate_t *, size_t);
extern double __wcstod_internal(const wchar_t *, wchar_t * *, int);
extern float __wcstof_internal(const wchar_t *, wchar_t * *, int);
extern long int __wcstol_internal(const wchar_t *, wchar_t * *, int, int);
extern long double __wcstold_internal(const wchar_t *, wchar_t * *, int, int);
extern unsigned long int __wcstoul_internal(const wchar_t *, wchar_t * *, int, int);
extern wchar_t *__wmemcpy_chk(wchar_t *, const wchar_t *, size_t, size_t);
extern wchar_t *__wmemmove_chk(wchar_t *, const wchar_t *, size_t, size_t);
extern wchar_t *__wmempcpy_chk(wchar_t *, const wchar_t *, size_t, size_t);
extern wchar_t *__wmemset_chk(wchar_t *, wchar_t, size_t, size_t);
extern int __wprintf_chk(int, const wchar_t *, ...);
extern wint_t btowc(int __c);
extern wint_t fgetwc(FILE *__stream);
extern wint_t fgetwc_unlocked(FILE *__stream);
extern wchar_t *fgetws(wchar_t *__ws, int __n, FILE *__stream);
extern wchar_t *fgetws_unlocked(wchar_t *__ws, int __n, FILE *__stream);
extern wint_t fputwc(wchar_t __wc, FILE *__stream);
extern wint_t fputwc_unlocked(wchar_t __wc, FILE *__stream);
extern int fputws(const wchar_t *__ws, FILE *__stream);
extern int fputws_unlocked(const wchar_t *__ws, FILE *__stream);
extern int fwide(FILE *__fp, int __mode);
extern int fwprintf(FILE *__stream, const wchar_t *__format, ...);
extern int fwscanf(FILE *__stream, const wchar_t *__format, ...);
extern wint_t getwc(FILE *__stream);
extern wint_t getwc_unlocked(void);
extern wint_t getwchar(void);
extern size_t mbrlen(const char *__s, size_t __n, mbstate_t *__ps);
extern size_t mbrtowc(wchar_t *__pwc, const char *__s, size_t __n, mbstate_t *__ps);
extern FILE *open_wmemstream(wchar_t *__bufloc, size_t *__sizeloc);
extern wint_t putwc(wchar_t __wc, FILE *__stream);
extern wint_t putwc_unlocked(wchar_t __wc, FILE *__stream);
extern wint_t putwchar(wchar_t __wc);
extern wint_t putwchar_unlocked(wchar_t __wc);
extern int swprintf(wchar_t *__s, size_t __n, const wchar_t *__format,
14 Base Libraries

extern float wcstof(const wchar_t * __nptr, wchar_t **__endptr);
extern wchar_t *wcstok(wchar_t * __s, const wchar_t * __delim,
    wchar_t **__ptr);
extern long int wcstol(const wchar_t * __nptr, wchar_t * __endptr,
    int __base);
extern double wcstold(const wchar_t * __nptr, wchar_t * __endptr);
extern long int wcstoll(const wchar_t * __nptr, wchar_t * __endptr,
    int __base);
extern long int wcstouq(const wchar_t * __nptr, wchar_t * __endptr,
    int __base);
extern unsigned long int wcstoul(const wchar_t * __nptr,
    wchar_t * __endptr, int __base);
extern unsigned long int wcstoull(const wchar_t * __nptr,
    wchar_t * __endptr, int __base);
extern wchar_t *wcscs(const wchar_t * __haystack, const wchar_t * __needle);
extern int wcswidth(const wchar_t * __s, size_t __n);
extern size_t wcsxfrm(wchar_t * __s1, const wchar_t * __s2,
    size_t __n);
extern size_t wcsxfrm_l(const wchar_t * ws1, const wchar_t * ws2,
    size_t n, locale_t locale);
extern int wctob(wint_t __c);
extern int wcwidth(wchar_t __c);
extern wchar_t *wmemchr(const wchar_t * __s, wchar_t __c, size_t __n);
extern int wmemcmp(const wchar_t * __s1, const wchar_t * __s2,
    size_t __n);
extern wchar_t *wmemcpy(wchar_t * __s1, const wchar_t * __s2,
    size_t __n);
extern wchar_t *wmemmove(wchar_t * __s1, const wchar_t * __s2,
    size_t __n);
extern wchar_t *wmemset(wchar_t * __s, wchar_t __c, size_t __n);
extern int wprintf(const wchar_t * __format, ...);
extern int wscanf(const wchar_t * __format, ...);

14.4.108 wctype.h

extern int iswalnum(wint_t __wc);
extern int iswalnum_l(wint_t __wc, locale_t locale);
extern int iswalpha(wint_t __wc);
extern int iswalpha_l(wint_t __wc, locale_t locale);
extern int iswblank(wint_t __wc);
extern int iswblank_l(wint_t __wc, locale_t locale);
extern int iswcntrl(wint_t __wc);
extern int iswcntrl_l(wint_t __wc, locale_t locale);
extern int iswctype(wint_t __wc, wctype_t __desc);
extern int iswctype_l(wint_t __wc, locale_t locale);
extern int iswdigit(wint_t __wc);
extern int iswdigit_l(wint_t __wc, locale_t locale);
extern int iswgraph(wint_t __wc);
extern int iswgraph_l(wint_t __wc, locale_t locale);
extern int iswlower(wint_t __wc);
extern int iswlower_l(wint_t __wc, locale_t locale);
extern int iswprint(wint_t __wc);
extern int iswprint_l(wint_t __wc, locale_t locale);
extern int iswupper(wint_t __wc);
extern int iswupper_l(wint_t __wc, locale_t locale);
extern int iswxdigit(wint_t __wc);
extern int iswxdigit_l(wint_t __wc, locale_t locale);
extern int iswupper(wint_t __wc);
extern int iswupper_l(wint_t __wc, locale_t locale);
extern int iswxdigit(wint_t __wc);
extern int iswxdigit_l(wint_t __wc, locale_t locale);
extern int iswpunct(wint_t __wc);
extern int iswpunct_l(wint_t wc, locale_t locale);
extern int iswspace(wint_t __wc);
extern int iswspace_l(wint_t wc, locale_t locale);
extern int iswupper(wint_t __wc);
extern int iswupper_l(wint_t wc, locale_t locale);
extern int iswxdigit(wint_t __wc);
extern int iswxdigit_l(wint_t wc, locale_t locale);
extern wint_t towctrans(wint_t __wc, wctrans_t __desc);
extern wint_t towctrans_l(wint_t wc, wctrans_t desc, locale_t locale);
extern wint_t towlower(wint_t __wc);
extern wint_t towlower_l(wint_t wc, locale_t locale);
extern wint_t towupper(wint_t __wc);
extern wint_t towupper_l(wint_t wc, locale_t locale);
extern wctrans_t wctrans(const char *__property);
extern size_t wctrans_l(const char *charclass, locale_t locale);
extern wctype_t wctype(const char *__property);
extern size_t wctype_l(const char *property, locale_t locale);

14.4.109 wordexp.h

enum {
    WRDE_DOOFFS = 1,
    WRDE_APPEND = 2,
    WRDE_NOCMD = 4,
    WRDE_REUSE = 8,
    WRDE_SHOWERR = 16,
    WRDE_UNDEF = 32
};

typedef struct {
    size_t we_wordc;
    char **we_wordv;
    size_t we_offs;
} wordexp_t;

enum {
    WRDE_NOSYS = -1,
    WRDE_NOSPACE = 1,
    WRDE_BADCHAR = 2,
    WRDE_BADVAL = 3,
    WRDE_CMDSUB = 4,
    WRDE_SYNTAX = 5
};

extern int wordexp(const char *__words, wordexp_t * __pwordexp,
                    int __flags);
extern void wordfree(wordexp_t * __wordexp);

14.5 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 14.3 shall behave as described in the referenced base
document.
_IO_feof

**Name**
_IO_feof — alias for feof

**Synopsis**

```c
int _IO_feof(_IO_FILE * __fp);
```

**Description**

_IO_feof() tests the end-of-file indicator for the stream pointed to by __fp, returning a non-zero value if it is set.

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

_EOF_getc

**Name**
_EOF_getc — alias for getc

**Synopsis**

```c
int _IO_getc(_IO_FILE * __fp);
```

**Description**

_EOF_getc() reads the next character from __fp and returns it as an unsigned char cast to an int, or EOF on end-of-file or error.

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

_EOF_putc

**Name**
_EOF_putc — alias for putc

**Synopsis**

```c
int _IO_putc(int __c, _IO_FILE * __fp);
```

**Description**

_EOF_putc() writes the character __c, cast to an unsigned char, to __fp.

_EOF_putc() is not in the source standard; it is only in the binary standard.
_IO_puts

Name
.IO_puts — alias for puts

Synopsis
int _IO_puts(const char * __c);

Description
_IO_puts() writes the string __s and a trailing newline to stdout.
_IO_puts() is not in the source standard; it is only in the binary standard.

__assert_fail

Name
__assert_fail — abort the program after false assertion

Synopsis
void __assert_fail(const char * assertion, const char * file, unsigned int line, const char * function);

Description
The __assert_fail() function is used to implement the assert() interface of
POSIX 1003.1-2008 (ISO/IEC 9945-2009). The __assert_fail() function shall print
the given file filename, line line number, function function name and a message
on the standard error stream in an unspecified format, and abort program execution via
the abort() function. For example:
a.c:10: foobar: Assertion a == b failed.
If function is NULL, __assert_fail() shall omit information about the function.
assertion, file, and line shall be non-NULL.
The __assert_fail() function is not in the source standard; it is only in the binary
standard. The assert() interface is not in the binary standard; it is only in the source
standard. The assert() may be implemented as a macro.
__chk_fail

Name
__chk_fail — terminate a function in case of buffer overflow

Synopsis
void __chk_fail(void);

Description
The interface __chk_fail() shall abort the function that called it with a message that
a buffer overflow has been detected. The program that called the function shall then
exit.

The __chk_fail() function is not in the source standard; it is only in the binary stan-
dard.

Application Usage (informative)
The interface __chk_fail() does not check for a buffer overflow itself. It merely re-
ports one when invoked.

__confstr_chk

Name
__confstr_chk — get configuration dependent string variables, with buffer
overflow checking

Synopsis
#include <unistd.h>
size_t __confstr_chk(int name, char * buf, size_t len, size_t buflen);

Description
The interface __confstr_chk() shall function in the same way as the interface conf-
str(), except that __confstr_chk() shall check for buffer overflow before comput-
ing a result. If an overflow is anticipated, the function shall abort and the program call-
ing it shall exit.

The parameter buflen specifies the size of the buffer buf. If len exceeds buflen, the
function shall abort, and the program calling it shall exit.

The __confstr_chk() function is not in the source standard; it is only in the binary
standard.
__ctype_b_loc

Name
__ctype_b_loc — accessor function for __ctype_b array for ctype functions

Synopsis
#include <ctype.h>
const unsigned short * * __ctype_b_loc (void);

Description
The __ctype_b_loc() function shall return a pointer into an array of characters in the current locale that contains characteristics for each character in the current character set. The array shall contain a total of 384 characters, and can be indexed with any signed or unsigned char (i.e. with an index value between -128 and 255). If the application is multithreaded, the array shall be local to the current thread.

This interface is not in the source standard; it is only in the binary standard.

Return Value
The __ctype_b_loc() function shall return a pointer to the array of characters to be used for the ctype() family of functions (see <ctype.h>).

__ctype_get_mb_cur_max

Name
__ctype_get_mb_cur_max — maximum length of a multibyte character in the current locale

Synopsis
size_t __ctype_get_mb_cur_max(void);

Description
__ctype_get_mb_cur_max() returns the maximum length of a multibyte character in the current locale.
__ctype_get_mb_cur_max() is not in the source standard; it is only in the binary standard.


__ctype_tolower_loc

Name
__ctype_tolower_loc — accessor function for __ctype_b_tolower array for ctype
tolower() function

Synopsis
#include <ctype.h>
int32_t ** __ctype_tolower_loc(void);

Description
The __ctype_tolower_loc() function shall return a pointer into an array of charac-
ters in the current locale that contains lower case equivalents for each character in the
current character set. The array shall contain a total of 384 characters, and can be in-
dexed with any signed or unsigned char (i.e. with an index value between -128 and
255). If the application is multithreaded, the array shall be local to the current thread.

This interface is not in the source standard; it is only in the binary standard.

Return Value
The __ctype_tolower_loc() function shall return a pointer to the array of characters
to be used for the ctype() family of functions (see <ctype.h>.

__ctype_toupper_loc

Name
__ctype_toupper_loc — accessor function for __ctype_b_toupper() array for ctype
toupper() function

Synopsis
#include <ctype.h>
int32_t ** __ctype_toupper_loc(void);

Description
The __ctype_toupper_loc() function shall return a pointer into an array of charac-
ters in the current locale that contains upper case equivalents for each character in the
current character set. The array shall contain a total of 384 characters, and can be in-
dexed with any signed or unsigned char (i.e. with an index value between -128 and
255). If the application is multithreaded, the array shall be local to the current thread.

This interface is not in the source standard; it is only in the binary standard.

Return Value
The __ctype_toupper_loc() function shall return a pointer to the array of characters
to be used for the ctype() family of functions (see <ctype.h>).
__cxa_atexit

**Name**

__cxa_atexit — register a function to be called by exit or when a shared library is unloaded

**Synopsis**

```c
int __cxa_atexit(void (*func) (void *), void * arg, void * dso_handle);
```

**Description**

As described in the Itanium™ C++ ABI, __cxa_atexit() registers a destructor function to be called by exit() or when a shared library is unloaded. When a shared library is unloaded, any destructor function associated with that shared library, identified by dso_handle, shall be called with the single argument arg, and then that function shall be removed, or marked as complete, from the list of functions to run at exit(). On a call to exit(), any remaining functions registered shall be called with the single argument arg. Destructor functions shall always be called in the reverse order to their registration (i.e. the most recently registered function shall be called first).

The __cxa_atexit() function is used to implement atexit(), as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009). Calling atexit(func) from the statically linked part of an application shall be equivalent to __cxa_atexit(func, NULL, NULL).

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

**Note:** atexit() is not in the binary standard; it is only in the source standard.
__cxa_finalize

Name
__cxa_finalize — call destructors of global (or local static) C++ objects and exit
functions registered with atexit

Synopsis
void __cxa_finalize(void * d);

Description
As described in the Itanium® C++ ABI, the C runtime library shall maintain a list of ter-
mination function entries containing the following information:

- A pointer to a termination function.
- An operand to be passed to the function.
- A handle identifying the home shared library of the entry.

The list is populated by entries of two kinds:
- Destructors of global (or local static) C++ objects that require destruction on exit.
- Functions registered by the user with atexit().

In the former case an entry consists of a pointer to the destructor, a pointer to the corre-
sponding object and a handle for the home shared library of the object. In the latter case
the pointer to the function is the pointer passed to atexit(), while the other pointers are
NULL.

When __cxa_finalize(d) is called, it shall walk the termination function list, calling each
in turn if d matches the handle of the termination function entry. If d is NULL, it shall
call all the termination funtions. Multiple calls to __cxa_finalize shall not result in call-
ing termination function entries multiple times; the implementation may either remove
entries or mark them finished. The termination functions shall always be called in the re-
verse order of their registration (i.e., the most recently registered function shall be called
first).

An application shall not call __cxa_finalize() directly. The implementation shall arrange
for__cxa_finalize() to be called during early shared library unload (e.g. dlclose()) with a
handle to the shared library. When the main program calls exit, the implementation shall
cause any remaining __cxa_atexit-registered functions to be called, either by calling
__cxa_finalize(NULL), or by walking the registration list itself.

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

__daylight

Name
__daylight — external daylight savings time flag

Synopsis
int __daylight;

Description
The external variable __daylight shall implement the daylight savings time flag day-
light as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009). __daylight has the
same specification as daylight.
__environ

Name
__environ — alias for environ - user environment

Synopsis
extern char **__environ;

Description
The external variable __environ shall implement the environment variable environ as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009). __environ has the same specification as environ.

__errno_location

Name
__errno_location — address of errno variable

Synopsis
int *__errno_location(void);

Description
The __errno_location() function shall return the address of the errno variable for the current thread.
__errno_location() is not in the source standard; it is only in the binary standard.

__fgets_chk

Name
__fgets_chk — string input, with buffer overflow checking

Synopsis
#include <stdio.h>
char *__fgets_chk(char * s, size_t size, int strsize, FILE * stream);

Description
The interface __fgets_chk() shall function in the same way as the interface fgets(), except that __fgets_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.
The parameter strsize specifies the size of the object pointed to by stream.
The __fgets_chk() function is not in the source standard; it is only in the binary standard.
__fgets_unlocked_chk

Name
__fgets_unlocked_chk — non-locking string input, with buffer overflow checking

Synopsis
#include <stdio.h>
char * __fgets_unlocked_chk(char * s, size_t size, int strsize, FILE * stream);

Description
The interface __fgets_unlocked_chk() shall function in the same way as the interface fgets_unlocked(), except that __fgets_unlocked_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter strsize specifies the size of the object pointed to by stream.

The __fgets_unlocked_chk() function is not in the source standard; it is only in the binary standard.

__fgetws_chk

Name
__fgetws_chk — read a wide-character string from a FILE stream, with buffer overflow checking

Synopsis
#include <wchar.h>
wchar_t * __fgetws_chk(wchar_t * ws, size_t size, int strsize, FILE * stream);

Description
The interface __fgetws_chk() shall function in the same way as the interface fgets(), except that __fgetws_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter strsize specifies the size of the object pointed to by stream.

The __fgetws_chk() function is not in the source standard; it is only in the binary standard.
__fgetws_unlocked_chk

**Name**

__fgetws_unlocked_chk — read a wide-character string from a FILE stream in a non-locking manner, with stack checking

**Synopsis**

```c
#include <wchar.h>
wchar_t * __fgetws_unlocked_chk (wchar_t * ws, size_t ssize, int n, FILE * stream);
```

**Description**

The interface __fgetws_unlocked_chk() shall function in the same way as the interface fgetws_unlocked(), except that __fgetws_unlocked_chk() shall check for stack overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter ssize specifies the size of the object pointed to by stream.

The __fgetws_unlocked_chk() function is not in the source standard; it is only in the binary standard.

__fpending

**Name**

__fpending — returns in bytes the amount of output pending on a stream

**Synopsis**

```c
size_t __fpending (FILE * stream);
```

**Description**

__fpending() returns the amount of output in bytes pending on a stream.

__fpending() is not in the source standard; it is only in the binary standard.
__fprintf_chk

Name
__fprintf_chk — convert formatted output, with stack checking

Synopsis
#include <stdio.h>
int __fprintf_chk(FILE * stream, int flag, const char * format);

Description
The interface __fprintf_chk() shall function in the same way as the interface fprintf(), except that __fprintf_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The __fprintf_chk() function is not in the source standard; it is only in the binary standard.

__fwprintf_chk

Name
__fwprintf_chk — convert formatted wide-character output, with stack checking

Synopsis
#include <wchar.h>
int __fwprintf_chk(FILE * stream, int flag, const wchar_t * format);

Description
The interface __fwprintf_chk() shall function in the same way as the interface fwprintf(), except that __fwprintf_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The __fwprintf_chk() function is not in the source standard; it is only in the binary standard.

__fxstatat

Name
__fxstatat — get file status relative to directory file descriptor

Synopsis
#include <fcntl.h>
#include <sys/stat.h>
int __fxstatat(int ver, int dirfd, const char * path, struct stat * stat_buf, int flags);

Description
The __fxstatat() function shall implement the fstatat() function. The behavior of __fxstatat() for values of ver other than _STAT_VER is undefined. See Data Definitions in the architecture specific part of this specification for the correct value of _STAT_VER.


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

Note: The fstatat() function is not in the binary standard; it is only in the source standard.

__fxstatat64, fstatat64

Name
__fxstatat64, fstatat64 — get file status relative to a directory file descriptor (Large File Support)

Synopsis
#include <fcntl.h>
#include <sys/stat.h>
int __fxstatat64(int ver, int dirfd, const char * path, struct stat64 * stat_buf, int flags);
int fstatat64(int dirfd, const char * file, struct stat64 * buf, int flag);

Description
fstatat64() is a large-file version of the fstatat() function as defined in POSIX 1003.1-2008 (ISO/IEC 9945-2009). It differs from fstatat() only in that the buf parameter refers to a large-file version of the stat structure.

The __fxstatat64() function shall implement the fstatat64() function. The behavior of __fxstatat64() for values of ver other than _STAT_VER is undefined. See Data Definitions in the architecture specific part of this specification for the correct value of _STAT_VER.

__fxstatat64(_STAT_VER, dirfd, stat_buf, flags) shall behave as fstatat64(dirfd, stat_buf, flags)

__fxstatat64() is not in the source standard; it is only in the binary standard.

Note: The fstatat64() function is not in the binary standard; it is only in the source standard.
__getcwd_chk

Name
__getcwd_chk — get current working directory, with buffer overflow checking

Synopsis

#include <unistd.h>
char *__getcwd_chk(char * buf, size_t len, size_t buflen);

Description

The interface __getcwd_chk() shall function in the same way as the interface getcwd(), except that __getcwd_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter buflen specifies the size of the buffer buf. If len exceeds buflen, the function shall abort, and the program calling it shall exit.

The __getcwd_chk() function is not in the source standard; it is only in the binary standard.

__getgroups_chk

Name
__getgroups_chk — get list of supplementary group IDs, with buffer overflow checking

Synopsis

#include <unistd.h>
int __getgroups_chk(int size, gid_t * list, size_t listlen);

Description

The interface __getgroups_chk() shall function in the same way as the interface getgroups(), except that __getgroups_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter listlen specifies the size in bytes of the object list.

The __getgroups_chk() function is not in the source standard; it is only in the binary standard.
__gethostname_chk

Name
__gethostname_chk — get host name, with buffer overflow checking

Synopsis
#include <unistd.h>
int __gethostname_chk(char * buf, size_t buflen, size_t maxlen);

Description
The interface __gethostname_chk() shall function in the same way as the interface gethostname(), except that __gethostname_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter buflen specifies the size of the buffer buf. If buflen exceeds maxlen, the function shall abort, and the program calling it shall exit.

The __gethostname_chk() function is not in the source standard; it is only in the binary standard.

__getlogin_r_chk

Name
__getlogin_r_chk — get user name, with buffer overflow checking (reentrant)

Synopsis
#include <unistd.h>
int __getlogin_r_chk(char * buf, size_t buflen, size_t maxlen);

Description
The interface __getlogin_r_chk() shall function in the same way as the interface getlogin_r(), except that __getlogin_r_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter buflen specifies the size of the buffer buf. If buflen exceeds maxlen, the function shall abort, and the program calling it shall exit.

The __getlogin_r_chk() function is not in the source standard; it is only in the binary standard.
__getpagesize

Name
__getpagesize — alias for getpagesize - get current page size

Synopsis
int __getpagesize(void);

Description
__getpagesize() is an alias for getpagesize() - get current page size.
__getpagesize() has the same specification as getpagesize().
__getpagesize() is not in the source standard; it is only in the binary standard.

__getpgid

Name
__getpgid — get the process group id

Synopsis
pid_t __getpgid(pid_t pid);

Description
__getpgid() has the same specification as getpgid().
__getpgid() is not in the source standard; it is only in the binary standard.

__h_errno_location

Name
__h_errno_location — address of h_errno variable

Synopsis
int * __h_errno_location(void);

Description
__h_errno_location() returns the address of the h_errno variable, where h_errno is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009).
__h_errno_location() is not in the source standard; it is only in the binary standard. Note that h_errno itself is only in the source standard; it is not in the binary standard.
__isinf

Name
__isinf — test for infinity

Synopsis
int __isinf(double arg);

Description
__isinf() has the same specification as isinf() in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except that the argument type for __isinf() is known to be double.
__isinf() is not in the source standard; it is only in the binary standard.

__isinff

Name
__isinff — test for infinity

Synopsis
int __isinff(float arg);

Description
__isinff() has the same specification as isinf() in POSIX 1003.1-2008 (ISO/IEC 9945-2009) except that the argument type for __isinff() is known to be float.
__isinff() is not in the source standard; it is only in the binary standard.

__isinfl

Name
__isinfl — test for infinity

Synopsis
int __isinfl(long double arg);

Description
__isinfl() has the same specification as isinf() in the POSIX 1003.1-2008 (ISO/IEC 9945-2009), except that the argument type for __isinfl() is known to be long double.
__isinfl() is not in the source standard; it is only in the binary standard.
__isnan

Name
__isnan — test for infinity

Synopsis
int __isnan(double arg);

Description
__isnan() has the same specification as isnan() in POSIX.1-2008 (ISO/IEC 9945-2009), except that the argument type for __isnan() is known to be double.
__isnan() is not in the source standard; it is only in the binary standard.

__isnanf

Name
__isnanf — test for infinity

Synopsis
int __isnanf(float arg);

Description
__isnanf() has the same specification as isnan() in POSIX.1-2008 (ISO/IEC 9945-2009), except that the argument type for __isnanf() is known to be float.
__isnanf() is not in the source standard; it is only in the binary standard.

__isnanl

Name
__isnanl — test for infinity

Synopsis
int __isnanl(long double arg);

Description
__isnanl() has the same specification as isnan() in POSIX.1-2008 (ISO/IEC 9945-2009), except that the argument type for __isnanl() is known to be long double.
__isnanl() is not in the source standard; it is only in the binary standard.
__libc_current_sigrtmax

Name
__libc_current_sigrtmax — return number of available real-time signal with
lowest priority

Synopsis
int __libc_current_sigrtmax(void);

Description
__libc_current_sigrtmax() returns the number of an available real-time signal
with the lowest priority.
__libc_current_sigrtmax() is not in the source standard; it is only in the binary
standard.

__libc_current_sigrtmin

Name
__libc_current_sigrtmin — return number of available real-time signal with
highest priority

Synopsis
int __libc_current_sigrtmin(void);

Description
__libc_current_sigrtmin() returns the number of an available real-time signal
with the highest priority.
__libc_current_sigrtmin() is not in the source standard; it is only in the binary
standard.
__libc_start_main

Name

__libc_start_main — initialization routine

Synopsis

int __libc_start_main(int (*main) (int, char **, char **), int argc, char ** ubp_av, void (*init) (void), void (*fini) (void), void (*rtld_fini) (void), void (*stack_end));

Description

The __libc_start_main() function shall perform any necessary initialization of the execution environment, call the main function with appropriate arguments, and handle the return from main(). If the main() function returns, the return value shall be passed to the exit() function.

Note: While this specification is intended to be implementation independent, process and library initialization may include:

• performing any necessary security checks if the effective user ID is not the same as the real user ID.
• initialize the threading subsystem.
• registering the rtld_fini to release resources when this dynamic shared object exits (or is unloaded).
• registering the fini handler to run at program exit.
• calling the initializer function (*init()).
• calling main() with appropriate arguments.
• calling exit() with the return value from main().

This list is an example only.

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

See Also

The section on Process Initialization in each of the architecture specific parts of the LSB Core Specification.
___mbsnrtowcs_chk

Name

___mbsnrtowcs_chk — convert a multibyte string to a wide-character string, with buffer overflow checking

Synopsis

#include <wchar.h>
size_t __mbsnrtowcs_chk(wchar_t * dest, const char ** src, size_t nmemb, size_t len, mbstate_t * ps, size_t destlen);

Description

The interface ___mbsnrtowcs_chk() shall function in the same way as the interface mbsnrtowcs(), except that ___mbsnrtowcs_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object dest. If len exceeds destlen, the function shall abort, and the program calling it shall exit.

The ___mbsnrtowcs_chk() function is not in the source standard; it is only in the binary standard.

___mbsrtowcs_chk

Name

___mbsrtowcs_chk — convert a multibyte string to a wide-character string, with buffer overflow checking

Synopsis

#include <wchar.h>
size_t __mbsrtowcs_chk(wchar_t * dest, const char * * src, size_t len, mbstate_t * ps, size_t destlen);

Description

The interface ___mbsrtowcs_chk() shall function in the same way as the interface mbsrtowcs(), except that ___mbsrtowcs_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object dest. If len exceeds destlen, the function shall abort, and the program calling it shall exit.

The ___mbsrtowcs_chk() function is not in the source standard; it is only in the binary standard.
__mbstowcs_chk

Name

__mbstowcs_chk — convert a multibyte string to a wide-character string, with
buffer overflow checking

Synopsis

#include <stdlib.h>
size_t __mbstowcs_chk(wchar_t * dest, const char * src, size_t len,
size_t destlen);

Description

The interface __mbstowcs_chk() shall function in the same way as the interface mb-
stowcs(), except that __mbstowcs_chk() shall check for buffer overflow before
computing a result. If an overflow is anticipated, the function shall abort and the pro-
gram calling it shall exit.

The parameter destlen specifies the size of the object dest. If len exceeds destlen,
the function shall abort, and the program calling it shall exit.

The __mbstowcs_chk() function is not in the source standard; it is only in the binary
standard.

__memcpy_chk

Name

__memcpy_chk — copy memory area, with buffer overflow checking

Synopsis

#include <string.h>
void * __memcpy_chk(void * dest, const void * src, size_t len,
size_t destlen);

Description

The interface __memcpy_chk() shall function in the same way as the interface mem-
cpy(), except that __memcpy_chk() shall check for buffer overflow before computing
a result. If an overflow is anticipated, the function shall abort and the program calling it
shall exit.

The parameter destlen specifies the size of the object dest. If len exceeds destlen,
the function shall abort, and the program calling it shall exit.

The __memcpy_chk() function is not in the source standard; it is only in the binary
standard.
__memmove_chk

Name
__memmove_chk — copy memory area, with buffer overflow checking

Synopsis
#include <string.h>
void * __memmove_chk(void * dest, const void * src, size_t len, size_t destlen);

Description
The interface __memmove_chk() shall function in the same way as the interface memmove(), except that __memmove_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object dest. If len exceeds destlen, the function shall abort, and the program calling it shall exit.

The __memmove_chk() function is not in the source standard; it is only in the binary standard.

__mempcpy

Name
__mempcpy — copy given number of bytes of source to destination

Synopsis
#include <string.h>
void * __mempcpy(void * restrict dest, const void * restrict src, size_t n);

Description
__mempcpy() copies n bytes of src to dest, returning a pointer to the byte after the last written byte.

If copying takes place between objects that overlap, the behavior is undefined.
If either dest or src is a null pointer, the behavior is undefined.
If n is 0 and the other parameters are valid, the return value is dest.
__mempcpy() is not in the source standard; it is only in the binary standard.
__mempcpy_chk

Name
__mempcpy_chk — copy memory area, with buffer overflow checking

Synopsis
#include <string.h>
void * __mempcpy_chk(void * dest, const void * src, size_t len, size_t destlen);

Description
The interface __mempcpy_chk() shall function in the same way as the interface mempcpy(), except that __mempcpy_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object dest. If len exceeds destlen, the function shall abort, and the program calling it shall exit.

The __mempcpy_chk() function is not in the source standard; it is only in the binary standard.

__memset_chk

Name
__memset_chk — fill memory with a constant byte, using buffer overflow checking

Synopsis
#include <string.h>
void * __memset_chk(void * dest, int c, size_t len, size_t destlen);

Description
The interface __memset_chk() shall function in the same way as the interface memset(), except that __memset_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object dest. If len exceeds destlen, the function shall abort, and the program calling it shall exit.

The __memset_chk() function is not in the source standard; it is only in the binary standard.
__pread64_chk

Name
__pread64_chk — read from a file descriptor at a given offset, with buffer overflow checking

Synopsis
#include <unistd.h>
ssize_t __pread64_chk(int fd, void * buf, size_t nbytes, off64_t offset, size_t buflen);

Description
The interface __pread64_chk() shall function in the same way as the interface pread64(), except that __pread64_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter buflen specifies the size of the buffer buf. If nbytes exceeds buflen, the function shall abort, and the program calling it shall exit.

The __pread64_chk() function is not in the source standard; it is only in the binary standard.

__pread_chk

Name
__pread_chk — read from a file descriptor at a given offset, with buffer overflow checking

Synopsis
#include <unistd.h>
ssize_t __pread_chk(int fd, void * buf, size_t nbytes, off_t offset, size_t buflen);

Description
The interface __pread_chk() shall function in the same way as the interface pread(), except that __pread_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter buflen specifies the size of the buffer buf. If nbytes exceeds buflen, the function shall abort, and the program calling it shall exit.

The __pread_chk() function is not in the source standard; it is only in the binary standard.
__printf_chk

Name
__printf_chk — format and print data, with stack checking

Synopsis
#include <stdio.h>
int __printf_chk(int flag, const char * format);

Description
The interface __printf_chk() shall function in the same way as the interface printf(), except that __printf_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The __printf_chk() function is not in the source standard; it is only in the binary standard.

__rawmemchr

Name
__rawmemchr — scan memory

Synopsis
#include <string.h>
void * __rawmemchr(const void * s, int c);

Description
The __rawmemchr() function shall locate the first occurrence of c (converted to an unsigned char) in the object pointed to by s. If the byte does not occur in the object, then the behavior is undefined.

__rawmemchr() is a weak alias for rawmemchr(). It is similar to memchr(), but it has no length limit.
__rawmemchr() is not in the source standard; it is only in the binary standard.

Return Value
The __rawmemchr() function shall return a pointer to the located byte.
__read_chk

Name
__read_chk — read from a file descriptor, with buffer overflow checking

Synopsis
#include <unistd.h>
ssize_t __read_chk(int fd, void * buf, size_t nbytes, size_t buflen);

Description
The interface __read_chk() shall function in the same way as the interface read(), except that __read_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter buflen specifies the size of the buffer buf. If nbytes exceeds buflen, the function shall abort, and the program calling it shall exit.

The __read_chk() function is not in the source standard; it is only in the binary standard.

__readlink_chk

Name
__readlink_chk — display value of a symbolic link, with buffer overflow checking

Synopsis
#include <unistd.h>
ssize_t __readlink_chk(const char * path, char * buf, size_t len, size_t buflen);

Description
The interface __readlink_chk() shall function in the same way as the interface readlink(), except that __readlink_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter buflen specifies the size of the buffer buf. If len exceeds buflen, the function shall abort, and the program calling it shall exit.

The __readlink_chk() function is not in the source standard; it is only in the binary standard.
__realpath_chk

Name
__realpath_chk — return the canonicalized absolute pathname, with buffer overflow checking

Synopsis
#include <stdlib.h>
char * __realpath_chk(const char * path, char * resolved_path, size_t resolved_len);

Description
The interface __realpath_chk() shall function in the same way as the interface realpath(), except that __realpath_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter resolved_len specifies the size of the string resolved_path. If resolved_len is less than PATH_MAX, then the function shall abort, and the program calling it shall exit.

The __realpath_chk() function is not in the source standard; it is only in the binary standard.

__recv_chk

Name
__recv_chk — receive a message from a socket, with buffer overflow checking

Synopsis
#include <sys/socket.h>
ssize_t __recv_chk(int fd, void * buf, size_t len, size_t buflen, int flag);

Description
The interface __recv_chk() shall function in the same way as the interface recv(), except that __recv_chk() shall check for buffer overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the buffer, parameter values, and so on.

The parameter buflen specifies the size of the buffer buf. If len exceeds buflen, the function shall abort, and the program calling it shall exit.

The __recv_chk() function is not in the source standard; it is only in the binary standard.
__recvfrom_chk

Name
__recvfrom_chk — receive a message from a socket, with buffer overflow checking

Synopsis
#include <sys/socket.h>
ssize_t __recvfrom_chk(int fd, void * buf, size_t len, size_t buflen, int flag, struct sockaddr * from, socklen_t * fromlen);

Description
The interface __recvfrom_chk() shall function in the same way as the interface recvfrom(), except that __recvfrom_chk() shall check for buffer overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the buffer, parameter values, and so on.

The parameter buflen specifies the size of the buffer buf. If len exceeds buflen, the function shall abort, and the program calling it shall exit.

The __recvfrom_chk() function is not in the source standard; it is only in the binary standard.

__register_atfork

Name
__register_atfork — alias for register_atfork

Synopsis
int __register_atfork(void (*prepare) (void), void (*parent) (void), void (*child) (void), void *__dso_handle);

Description
__register_atfork() implements pthread_atfork() as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009). The additional parameter __dso_handle allows a shared object to pass in it's handle so that functions registered by __register_atfork() can be unregistered by the runtime when the shared object is unloaded.
__sigsetjmp

Name
__sigsetjmp — save stack context for non-local goto

Synopsis
int __sigsetjmp(jmp_buf env, int savemask);

Description
__sigsetjmp() has the same behavior as sigsetjmp() as specified by POSIX 1003.1-2008 (ISO/IEC 9945-2009).
__sigsetjmp() is not in the source standard; it is only in the binary standard.

__snprintf_chk

Name
__snprintf_chk — convert formatted output, with buffer overflow checking

Synopsis
#include <stdio.h>
int __snprintf_chk(char * str, size_t maxlen, int flag, size_t strlen, const char * format);

Description
The interface __snprintf_chk() shall function in the same way as the interface snprintf(), except that __snprintf_chk() shall check for buffer overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the buffer, parameter values, and so on.

The parameter strlen specifies the size of the buffer str. If strlen is less than maxlen, the function shall abort, and the program calling it shall exit.

The __snprintf_chk() function is not in the source standard; it is only in the binary standard.
__sprintf_chk

Name
__sprintf_chk — convert formatted output, with stack checking

Synopsis
#include <stdio.h>
int __sprintf_chk(char * str, int flag, size_t strlen, const char * format);

Description
The interface __sprintf_chk() shall function in the same way as the interface sprintf(), except that __sprintf_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The parameter strlen specifies the size of the string str. If strlen is zero, the function shall abort, and the program calling it shall exit.

The __sprintf_chk() function is not in the source standard; it is only in the binary standard.

__stack_chk_fail

Name
__stack_chk_fail — terminate a function in case of stack overflow

Synopsis
void __stack_chk_fail(void);

Description
The interface __stack_chk_fail() shall abort the function that called it with a message that a stack overflow has been detected. The program that called the function shall then exit.

The __stack_chk_fail() function is not in the source standard; it is only in the binary standard.

Application Usage (informative)
The interface __stack_chk_fail() does not check for a stack overflow itself. It merely reports one when invoked.
__stpcpy

Name
__stpcpy — alias for stpcpy

Synopsis
#include <string.h>
char * __stpcpy(char * dest, const char * src);

Description
The __stpcpy() function has the same specification as the stpcpy().
__stpcpy() is not in the source standard; it is only in the binary standard.

__stpcpy_chk

Name
__stpcpy_chk — copy a string returning a pointer to its end, with buffer overflow checking

Synopsis
#include <string.h>
char * __stpcpy_chk(char * dest, const char * src, size_t destlen);

Description
The interface __stpcpy_chk() shall function in the same way as the interface stpcpy(), except that __stpcpy_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by dest.
The __stpcpy_chk() function is not in the source standard; it is only in the binary standard.
__stpcpy_chk

Name
__stpcpy_chk — copy a fixed-size string, returning a pointer to its end, with buffer overflow checking

Synopsis
#include <string.h>
char *__stpcpy_chk(char * dest, const char * src, size_t n, size_t destlen);

Description
The interface __stpcpy_chk() shall function in the same way as the interface stpcpy(), except that __stpcpy_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by dest. If n exceeds destlen, the function shall abort, and the program calling it shall exit.

The __stpcpy_chk() function is not in the source standard; it is only in the binary standard.

__strcat_chk

Name
__strcat_chk — concatenate two strings, with buffer overflow checking

Synopsis
#include <string.h>
char *__strcat_chk(char * dest, const char * src, size_t destlen);

Description
The interface __strcat_chk() shall function in the same way as the interface strcat(), except that __strcat_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by dest.

The __strcat_chk() function is not in the source standard; it is only in the binary standard.
__strcpy_chk

Name
__strcpy_chk — copy a string, with buffer overflow checking

Synopsis
#include <string.h>
char * __strcpy_chk(char * dest, const char * src, size_t destlen);

Description
The interface __strcpy_chk() shall function in the same way as the interface strcpy(), except that __strcpy_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by dest.

The __strcpy_chk() function is not in the source standard; it is only in the binary standard.

__strdup

Name
__strdup — alias for strdup

Synopsis
char * __strdup(const char * string);

Description
__strdup() has the same specification as strdup().
__strdup() is not in the source standard; it is only in the binary standard.

__strncat_chk

Name
__strncat_chk — concatenate two strings, with buffer overflow checking

Synopsis
#include <string.h>
char * __strncat_chk(char * s1, const char * s2, size_t n, size_t s1len);

Description
The interface __strncat_chk() shall function in the same way as the interface strncat(), except that __strncat_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter s1len specifies the size of the object pointed to by s1.

The __strncat_chk() function is not in the source standard; it is only in the binary standard.
__strncpy_chk

Name
__strncpy_chk — copy a string, with buffer overflow checking

Synopsis
#include <string.h>
char * __strncpy_chk(char * s1, const char * s2, size_t n, size_t s1len);

Description
The interface __strncpy_chk() shall function in the same way as the interface
strncpy(), except that __strncpy_chk() shall check for buffer overflow before
computing a result. If an overflow is anticipated, the function shall abort and the pro-
gram calling it shall exit.

The parameter s1len specifies the size of the object pointed to by s1.

The __strncpy_chk() function is not in the source standard; it is only in the binary
standard.

__strtod_internal

Name
__strtod_internal — underlying function for strtod

Synopsis
double __strtod_internal(const char * __nptr, char * __endptr, int __group);

Description
__group shall be 0 or the behavior of __strtod_internal() is undefined.

__strtod_internal(__nptr, __endptr, 0)() has the same specification as
strtod(__nptr, __endptr)().

__strtod_internal() is not in the source standard; it is only in the binary standard.
__strtof_internal

Name
__strtof_internal — underlying function for strtof

Synopsis
float __strtof_internal(const char * __nptr, char * __endptr, int __group);

Description
__group shall be 0 or the behavior of __strtof_internal() is undefined.
__strtof_internal(__nptr, __endptr, 0)() has the same specification as
strtof(__nptr, __endptr)().
__strtof_internal() is not in the source standard; it is only in the binary standard.

__strtok_r

Name
__strtok_r — alias for strtok_r

Synopsis
char * __strtok_r(char * restrict s, const char * restrict delim,
char * * restrict save_ptr);

Description
__strtok_r() has the same specification as strtok_r().
__strtok_r() is not in the source standard; it is only in the binary standard.

__strtol_internal

Name
__strtol_internal — alias for strtol

Synopsis
long int __strtol_internal(const char * __nptr, char * __endptr, int __base, int __group);

Description
__group shall be 0 or the behavior of __strtol_internal() is undefined.
__strtol_internal(__nptr, __endptr, __base, 0) has the same specification as
strtol(__nptr, __endptr, __base).
__strtol_internal() is not in the source standard; it is only in the binary standard.
__strtold_internal

Name
__strtold_internal — underlying function for strtold

Synopsis
long double __strtold_internal(const char *__nptr, char **__endptr, int __group);

Description
__group shall be 0 or the behavior of __strtold_internal() is undefined.
__strtold_internal(__nptr, __endptr, 0) has the same specification as
strtold(__nptr, __endptr).
__strtold_internal() is not in the source standard; it is only in the binary standard.

__strtoll_internal

Name
__strtoll_internal — underlying function for strtoll

Synopsis
long long __strtoll_internal(const char *__nptr, char **__endptr, int __base, int __group);

Description
__group shall be 0 or the behavior of __strtoll_internal() is undefined.
__strtoll_internal(__nptr, __endptr, __base, 0) has the same specification as
strtoll(__nptr, __endptr, __base).
__strtoll_internal() is not in the source standard; it is only in the binary standard.

__strtoul_internal

Name
__strtoul_internal — underlying function for strtoul

Synopsis
unsigned long int __strtoul_internal(const char *__nptr, char **__endptr, int __base, int __group);

Description
__group shall be 0 or the behavior of __strtoul_internal() is undefined.
__strtoul_internal(__nptr, __endptr, __base, 0) has the same specification as
strtoul(__nptr, __endptr, __base).
__strtoul_internal() is not in the source standard; it is only in the binary standard.
**__strtoull_internal**

**Name**
__strtoull_internal — underlying function for strtoull

**Synopsis**

```c
unsigned long long __strtoull_internal(const char * __nptr, char * *__endptr, int __base, int __group);
```

**Description**

__group shall be 0 or the behavior of __strtoull_internal() is undefined.

__strtoull_internal(__nptr, __endptr, __base, 0) has the same specification as strtoull(__nptr, __endptr, __base).

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

**__swprintf_chk**

**Name**
__swprintf_chk — convert formatted wide-character output, with stack checking

**Synopsis**

```c
#include <wchar.h>
int __swprintf_chk(wchar_t * s, size_t n, int flag, size_t slen, const wchar_t * format);
```

**Description**

The interface __swprintf_chk() shall function in the same way as the interface swprintf(), except that __swprintf_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The parameter slen specifies the size of the object pointed to by s. If slen is less than maxlen, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The __swprintf_chk() function is not in the source standard; it is only in the binary standard.
__sysconf

Name
__sysconf — get configuration information at runtime

Synopsis
#include <unistd.h>
long __sysconf(int name);

Description
__sysconf() gets configuration information at runtime.
__sysconf() is weak alias to sysconf().
__sysconf() has the same specification as sysconf().
__sysconf() is not in the source standard; it is only in the binary standard.

__syslog_chk

Name
__syslog_chk — send messages to the system logger, with stack checking

Synopsis
#include <syslog.h>
void __syslog_chk(int priority, int flag, const char * format);

Description
The interface __syslog_chk() shall function in the same way as the interface syslog(), except that __syslog_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The __syslog_chk() function is not in the source standard; it is only in the binary standard.

__sysv_signal

Name
__sysv_signal — signal handling

Synopsis
__sighandler_t __sysv_signal(int sig, __sighandler_t handler);

Description
__sysv_signal() has the same behavior as signal() as specified by POSIX.1-2008 (ISO/IEC 9945-2009).
__sysv_signal() is not in the source standard; it is only in the binary standard.
__timezone

Name
__timezone — external variable containing timezone

Synopsis
long int __timezone;

Description
The external variable __timezone shall implement the timezone variable timezone as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009). __timezone has the same specification as timezone.

__ttyname_r_chk

Name
__ttyname_r_chk — return name of a terminal, with buffer overflow checking (reentrant)

Synopsis
#include <unistd.h>
int __ttyname_r_chk(int fd, char * buf, size_t buflen, size_t nreal);

Description
The interface __ttyname_r_chk() shall function in the same way as the interface ttyname_r(), except that __ttyname_r_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter buflen specifies the size of the object pointed to by buf. If buflen exceeds nreal, the function shall abort and the program calling it shall exit.

The __ttyname_r_chk() function is not in the source standard; it is only in the binary standard.

__tzname

Name
__tzname — external variable containing the timezone names

Synopsis
char * __tzname[2];

Description
The external variable __tzname shall implement the timezone name variable tzname as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009) function tzset(). __tzname has the same specification as tzname.
__vfprintf_chk

Name
__vfprintf_chk — convert formatted output, with stack checking

Synopsis
#include <stdio.h>
int __vfprintf_chk(FILE * fp, int flag, const char * format, va_list ap);

Description
The interface __vfprintf_chk() shall function in the same way as the interface vfprintf(), except that __vfprintf_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The __vfprintf_chk() function is not in the source standard; it is only in the binary standard.

__vfwprintf_chk

Name
__vfwprintf_chk — convert formatted wide-character output, with stack checking

Synopsis
#include <wchar.h>
int __vfwprintf_chk(FILE * fp, int flag, const wchar_t * format, va_list ap);

Description
The interface __vfwprintf_chk() shall function in the same way as the interface vfwprintf(), except that __vfwprintf_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The __vfwprintf_chk() function is not in the source standard; it is only in the binary standard.
__vprintf_chk

Name
__vprintf_chk — convert formatted output, with stack checking

Synopsis
#include <stdio.h>
int __vprintf_chk(int flag, const char * format, va_list ap);

Description
The interface __vprintf_chk() shall function in the same way as the interface
vprintf(), except that __vprintf_chk() shall check for stack overflow before com-
puting a result, depending on the value of the flag parameter. If an overflow is anticip-
ated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall
take in the form of checking the stack, parameter values, and so on.

The __vprintf_chk() function is not in the source standard; it is only in the binary
standard.

__vsnprintf_chk

Name
__vsnprintf_chk — convert formatted output, with stack checking

Synopsis
#include <stdio.h>
int __vsnprintf_chk(char * s, size_t maxlen, int flag, size_t slen,
const char * format, va_list args);

Description
The interface __vsnprintf_chk() shall function in the same way as the interface vs-
nprintf(), except that __vsnprintf_chk() shall check for stack overflow before com-
puting a result, depending on the value of the flag parameter. If an overflow is anticip-
ated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall
take in the form of checking the stack, parameter values, and so on.

The parameter slen specifies the size of the object pointed to by s. If slen is less than
maxlen, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall
take in the form of checking the stack, parameter values, and so on.

The __vsnprintf_chk() function is not in the source standard; it is only in the binary
standard.
__vsprintf_chk

Name
__vsprintf_chk — convert formatted output, with stack checking

Synopsis
#include <stdio.h>
int __vsprintf_chk(char * s, int flag, size_t slen, const char * format, va_list args);

Description
The interface __vsprintf_chk() shall function in the same way as the interface vsprintf(), except that __vsprintf_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The parameter slen specifies the size of the object pointed to by s. If its value is zero, the function shall abort and the program calling it shall exit.

The __vsprintf_chk() function is not in the source standard; it is only in the binary standard.

__vswprintf_chk

Name
__vswprintf_chk — convert formatted wide-character output, with stack checking

Synopsis
#include <wchar.h>
int __vswprintf_chk(wchar_t * s, size_t maxlen, int flag, size_t slen, const wchar_t * format, va_list args);

Description
The interface __vswprintf_chk() shall function in the same way as the interface vswprintf(), except that __vswprintf_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The parameter slen specifies the size of the object pointed to by s. If slen is less than maxlen, the function shall abort and the program calling it shall exit.

The __vswprintf_chk() function is not in the source standard; it is only in the binary standard.
__vsyslog_chk

Name
__vsyslog_chk — send messages to the system logger, with stack checking

Synopsis
#include <syslog.h>
void __vsyslog_chk(int priority, int flag, const char * format, va_list ap);

Description
The interface __vsyslog_chk() shall function in the same way as the interface syslog(), except that __vsyslog_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The __vsyslog_chk() function is not in the source standard; it is only in the binary standard.

__vwprintf_chk

Name
__vwprintf_chk — convert formatted wide-character output, with stack checking

Synopsis
#include <wchar.h>
int __vwprintf_chk(int flag, const wchar_t * format, va_list ap);

Description
The interface __vwprintf_chk() shall function in the same way as the interface vprintf(), except that __vwprintf_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The __vwprintf_chk() function is not in the source standard; it is only in the binary standard.
___wcpcpy_chk

Name

___wcpcpy_chk — copy a wide-character string, returning a pointer to its end, with buffer overflow checking

Synopsis

```
#include <wchar.h>
wchar_t * __wcpcpy_chk(wchar_t * dest, const wchar_t * src, size_t destlen);
```

Description

The interface ___wcpcpy_chk() shall function in the same way as the interface wcpncpy(), except that ___wcpcpy_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by dest.

The ___wcpcpy_chk() function is not in the source standard; it is only in the binary standard.

___wcpncpy_chk

Name

___wcpncpy_chk — copy a fixed-size string of wide characters, returning a pointer to its end, with buffer overflow checking

Synopsis

```
#include <wchar.h>
wchar_t * __wcpncpy_chk(wchar_t * dest, const wchar_t * src, size_t n, size_t destlen);
```

Description

The interface ___wcpncpy_chk() shall function in the same way as the interface wcpncpy(), except that ___wcpncpy_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by dest. If n exceeds destlen, the function shall abort and the program calling it shall exit.

The ___wcpncpy_chk() function is not in the source standard; it is only in the binary standard.
__wcrtomb_chk

Name
__wcrtomb_chk — convert a wide character to a multibyte sequence, with buffer
overflow checking

Synopsis
#include <wchar.h>
size_t __wcrtomb_chk(char * s, wchar_t wchar, mbstate_t * ps, size_t buflen);

Description
The interface __wcrtomb_chk() shall function in the same way as the interface
wcrtomb(), except that __wcrtomb_chk() shall check for buffer overflow before
computing a result. If an overflow is anticipated, the function shall abort and the pro-
gram calling it shall exit.

The parameter buflen specifies the size of the object pointed to by s. If it is less than
MB_CUR_MAX, then the function shall abort and the program calling it shall exit.

The __wcrtomb_chk() function is not in the source standard; it is only in the binary
standard.

__wcscat_chk

Name
__wcscat_chk — concatenate two wide-character strings, with buffer overflow
checking

Synopsis
#include <wchar.h>
wchar_t * __wcscat_chk(wchar_t * dest, const wchar_t * src, size_t destlen);

Description
The interface __wcscat_chk() shall function in the same way as the interface wc-
scat(), except that __wcscat_chk() shall check for buffer overflow before comput-
ing a result. If an overflow is anticipated, the function shall abort and the program call-
ing it shall exit.

The parameter destlen specifies the size of the object pointed to by dest.

The __wcscat_chk() function is not in the source standard; it is only in the binary
standard.
__wcscpy_chk

Name

__wcscpy_chk — copy a wide-character string, with buffer overflow checking

Synopsis

#include <wchar.h>

wchar_t * __wcscpy_chk(wchar_t * dest, const wchar_t * src, size_t n);

Description

The interface __wcscpy_chk() shall function in the same way as the interface wcscpy(), except that __wcscpy_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The __wcscpy_chk() function is not in the source standard; it is only in the binary standard.

__wcsncat_chk

Name

__wcsncat_chk — concatenate two wide-character strings, with buffer overflow checking

Synopsis

#include <wchar.h>

wchar_t * __wcsncat_chk(wchar_t * dest, const wchar_t * src, size_t n, size_t destlen);

Description

The interface __wcsncat_chk() shall function in the same way as the interface wcsncat(), except that __wcsncat_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by dest.

The __wcsncat_chk() function is not in the source standard; it is only in the binary standard.
__wcsncpy_chk

Name
__wcsncpy_chk — copy a fixed-size string of wide characters, with buffer overflow checking

Synopsis
#include <wchar.h>
wchar_t *__wcsncpy_chk(wchar_t * dest, const wchar_t * src, size_t n, size_t destlen);

Description
The interface __wcsncpy_chk() shall function in the same way as the interface wcsncpy(), except that __wcsncpy_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by dest. If len exceeds destlen, the function shall abort and the program calling it shall exit.

The __wcsncpy_chk() function is not in the source standard; it is only in the binary standard.

__wcnrtombs_chk

Name
__wcnrtombs_chk — convert a wide-character string to a multibyte string, with buffer overflow checking

Synopsis
#include <wchar.h>
size_t __wcnrtombs_chk(char * dest, const wchar_t * * src, size_t nwc, size_t len, mbstate_t * ps, size_t destlen);

Description
The interface __wcnrtombs_chk() shall function in the same way as the interface wcrtombs(), except that __wcnrtombs_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by dest. If len exceeds destlen, the function shall abort and the program calling it shall exit.

The __wcnrtombs_chk() function is not in the source standard; it is only in the binary standard.
__wcsrtombs_chk

Name

__wcsrtombs_chk — convert a wide-character string to a multibyte string, with buffer overflow checking

Synopsis

#include <wchar.h>
size_t __wcsrtombs_chk(char * dest, const wchar_t * * src, size_t len, mbstate_t * ps, size_t destlen);

Description

The interface __wcsrtombs_chk() shall function in the same way as the interface wcsrtombs(), except that __wcsrtombs_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by dest. If len exceeds destlen, the function shall abort and the program calling it shall exit.

The __wcsrtombs_chk() function is not in the source standard; it is only in the binary standard.

__wcstod_internal

Name

__wcstod_internal — underlying function for wcstod

Synopsis

double __wcstod_internal(const wchar_t * nptr, wchar_t * * endptr, int group);

Description

group shall be 0 or the behavior of __wcstod_internal() is undefined.

__wcstod_internal(nptr, endptr, 0) shall behave as wcstod(nptr, endptr) as specified by POSIX 1003.1-2008 (ISO/IEC 9945-2009).

__wcstod_internal() is not in the source standard; it is only in the binary standard.
__wcstof_internal

Name

__wcstof_internal — underlying function for wcstof

Synopsis

float __wcstof_internal(const wchar_t * nptr, wchar_t * * endptr, int group);

Description

group shall be 0 or the behavior of __wcstof_internal() is undefined.

__wcstof_internal(nptr, endptr, 0) shall behave as wcstof(nptr, endptr) as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009).

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

__wcstol_internal

Name

__wcstol_internal — underlying function for wcstol

Synopsis

long __wcstol_internal(const wchar_t * nptr, wchar_t * * endptr, int base, int group);

Description

group shall be 0 or the behavior of __wcstol_internal() is undefined.

__wcstol_internal(nptr, endptr, base, 0) shall behave as wcstol(nptr, endptr, base) as specified by POSIX 1003.1-2008 (ISO/IEC 9945-2009).

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

__wcstold_internal

Name

__wcstold_internal — underlying function for wcstold

Synopsis

long double __wcstold_internal(const wchar_t * nptr, wchar_t * * endptr, int group);

Description

group shall be 0 or the behavior of __wcstold_internal() is undefined.

__wcstold_internal(nptr, endptr, 0) shall behave as wcstold(nptr, endptr) as specified by POSIX 1003.1-2008 (ISO/IEC 9945-2009).

__wcstold_internal() is not in the source standard; it is only in the binary standard.
__wcstombs_chk

**Name**

__wcstombs_chk — convert a wide-character string to a multibyte string, with buffer overflow checking

**Synopsis**

```c
#include <stdlib.h>
size_t __wcstombs_chk(char * dest, const wchar_t * src, size_t len, size_t destlen);
```

**Description**

The interface __wcstombs_chk() shall function in the same way as the interface wcstombs(), except that __wcstombs_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by dest. If len exceeds destlen, the function shall abort and the program calling it shall exit.

The __wcstombs_chk() function is not in the source standard; it is only in the binary standard.

__wcstoul_internal

**Name**

__wcstoul_internal — underlying function for wcstoul

**Synopsis**

```c
unsigned long __wcstoul_internal(const wchar_t * restrict nptr, wchar_t * restrict endptr, int base, int group);
```

**Description**

group shall be 0 or the behavior of __wcstoul_internal() is undefined.

__wcstoul_internal(nptr, endptr, base, 0)() shall behave as wcstoul(nptr, endptr, base)() as specified by POSIX 1003.1-2008 (ISO/IEC 9945-2009).

__wcstoul_internal() is not in the source standard; it is only in the binary standard.
__wctomb_chk

Name
__wctomb_chk — convert a wide character to a multibyte sequence, with buffer overflow checking

Synopsis
#include <stdlib.h>
int __wctomb_chk(char * s, wchar_t wchar, size_t buflen);

Description
The interface __wctomb_chk() shall function in the same way as the interface wc_tomb(), except that __wctomb_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter buflen specifies the size of the object pointed to by s. If it is less than MB_CUR_MAX, then the function shall abort and the program calling it shall exit.

The __wctomb_chk() function is not in the source standard; it is only in the binary standard.

__wmemcpy_chk

Name
__wmemcpy_chk — copy an array of wide-characters, with buffer overflow checking

Synopsis
#include <wchar.h>
wchar_t * __wmemcpy_chk(wchar_t * s1, const wchar_t * s2, size_t n, size_t ns1);

Description
The interface __wmemcpy_chk() shall function in the same way as the interface wmemcpy(), except that __wmemcpy_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter ns1 specifies the size of the object pointed to by s1. If n exceeds ns1, the function shall abort and the program calling it shall exit.

The __wmemcpy_chk() function is not in the source standard; it is only in the binary standard.
__wmemmove_chk

Name
__wmemmove_chk — copy an array of wide-characters, with buffer overflow checking

Synopsis
#include <wchar.h>
wchar_t * __wmemmove_chk(wchar_t * s1, const wchar_t * s2, size_t n, size_t ns1);

Description
The interface __wmemmove_chk() shall function in the same way as the interface wmemmove(), except that __wmemmove_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter ns1 specifies the size of the object pointed to by s1. If n exceeds ns1, the function shall abort and the program calling it shall exit.

The __wmemmove_chk() function is not in the source standard; it is only in the binary standard.

__wmempcpy_chk

Name
__wmempcpy_chk — copy memory area, with buffer overflow checking

Synopsis
#include <wchar.h>
wchar_t * __wmempcpy_chk(wchar_t * s1, const wchar_t * s2, size_t n, size_t ns1);

Description
The interface __wmempcpy_chk() shall function in the same way as the interface wmempcpy(), except that __wmempcpy_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter ns1 specifies the size of the object pointed to by s1. If n exceeds ns1, the function shall abort and the program calling it shall exit.

The __wmempcpy_chk() function is not in the source standard; it is only in the binary standard.
__wmemset_chk

Name
__wmemset_chk — fill an array of wide-characters with a constant wide character, with buffer overflow checking

Synopsis
#include <wchar.h>
wchar_t * __wmemset_chk(wchar_t * s, wchar_t c, size_t n, size_t destlen);

Description
The interface __wmemset_chk() shall function in the same way as the interface wmemset(), except that __wmemset_chk() shall check for buffer overflow before computing a result. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

The parameter destlen specifies the size of the object pointed to by s. If n exceeds destlen, the function shall abort and the program calling it shall exit.

The __wmemset_chk() function is not in the source standard; it is only in the binary standard.

__wprintf_chk

Name
__wprintf_chk — convert formatted wide-character output, with stack checking

Synopsis
#include <wchar.h>
int __wprintf_chk(int flag, const wchar_t * format);

Description
The interface __wprintf_chk() shall function in the same way as the interface wprintf(), except that __wprintf_chk() shall check for stack overflow before computing a result, depending on the value of the flag parameter. If an overflow is anticipated, the function shall abort and the program calling it shall exit.

In general, the higher the value of flag, the more security measures this interface shall take in the form of checking the stack, parameter values, and so on.

The __wprintf_chk() function is not in the source standard; it is only in the binary standard.
__xmknod

Name
__xmknod — make a special file

Synopsis
#include <sys/stat.h>
int __xmknod(int ver, const char *path, mode_t mode, dev_t *dev);

Description
The __xmknod() function shall implement the mknod() interface. The behavior of __xmknod() for values of ver other than _MKNOD_VER is undefined. See Data Definitions in the architecture specific part of this specification for the correct value of _MKNOD_VER.
__xmknod(_MKNOD_VER, path, mode, dev) shall behave as mknod(path, mode, dev) as specified by POSIX 1003.1-2008 (ISO/IEC 9945-2009).

The __xmknod() function is not in the source standard; it is only in the binary standard.

__xmknodat

Name
__xmknodat — make a special file relative to a directory file descriptor

Synopsis
#include <sys/stat.h>
int __xmknodat(int ver, int dirfd, const char *path, mode_t path,
dev_t *dev);

Description
The __xmknodat() function shall implement the mknodat() function. The behavior of __xmknodat() for values of ver other than _MKNOD_VER is undefined. See Data Definitions in the architecture specific part of this specification for the correct value of _MKNOD_VER.
__xmknodat(_MKNOD_VER, dirfd, path, mode, dev) shall behave as mknodat(dirfd, path, mode, dev) as specified by POSIX 1003.1-2008 (ISO/IEC 9945-2009).

The __xmknodat() function is not in the source standard; it is only in the binary standard.

Note: The mknodat() function is not in the binary standard; it is only in the source standard.
__xpg_basename

Name
__xpg_basename — return the last component of a file name

Synopsis
#include <libgen.h>
char * __xpg_basename(const char * path);

Description
The __xpg_basename() function shall return a pointer to the final component of the
pathname named by path, as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009).
basename().
This function is not in the source standard, it is only in the binary standard.

Return Value

__xpg_sigpause

Name
__xpg_sigpause — remove a signal from the signal mask and suspend the thread

Synopsis
#include <signal.h>
int __xpg_sigpause(int sig);

Description
The __xpg_sigpause() function shall implement the sigpause() described in
This function is not in the source standard, it is only in the binary standard.

Return Value
__xpg_strerror_r

Name
__xpg_strerror_r — return string describing error number

Synopsis
#include <string.h>
int __xpg_strerror_r(int errnum, char * buf, size_t buflen);

Description
The __xpg_strerror_r() function shall map the error number in errnum to a locale-dependent error message string and shall return the string in the buffer pointed to by strerrorbuf, with length buflen, as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009) strerror_r().

This function is not in the source standard, it is only in the binary standard.

Return Value

__xstat

Name
__xstat — get File Status

Synopsis
#include <sys/stat.h>
#include <unistd.h>
int __xstat(int ver, const char * path, struct stat * stat_buf);
int __lxstat(int ver, const char * path, struct stat * stat_buf);
int __fxstat(int ver, int fildes, struct stat * stat_buf);

Description
The functions __xstat(), __lxstat(), and __fxstat() shall implement the functions stat(), lstat(), and fstat() respectively.

The behavior of these functions for values of ver other than _STAT_VER is undefined. See Data Definitions in the architecture specific part of this specification for the correct value of _STAT_VER.

__xstat(_STAT_VER, path, stat_buf) shall implement stat(path, stat_buf) as specified by POSIX 1003.1-2008 (ISO/IEC 9945-2009).

__lxstat(_STAT_VER, path, stat_buf) shall implement lstat(path, stat_buf) as specified by POSIX 1003.1-2008 (ISO/IEC 9945-2009).


__xstat(), __lxstat(), and __fxstat() are not in the source standard; they are only in the binary standard.

stat(), lstat(), and fstat() are not in the binary standard; they are only in the source standard.
__xstat64

Name

__xstat64 — get File Status

Synopsis

#define _LARGEFILE_SOURCE 1
#include <sys/stat.h>
#include <unistd.h>
int __xstat64(int ver, const char *path, struct stat64 *stat_buf);
int __lxstat64(int ver, const char *path, struct stat64 *stat_buf);
int __fxstat64(int ver, int fildes, struct stat64 *stat_buf);

Description

The functions __xstat64(), __lxstat64(), and __fxstat64() shall implement the functions stat64(), lstat64(), and fstat64() respectively.

The behavior of these functions for values of ver other than _STAT_VER is undefined. See Data Definitions in the architecture specific part of this specification for the correct value of _STAT_VER.

__xstat64(_STAT_VER, path, stat_buf) shall behave as stat64(path, stat_buf) as specified by Large File Support.

__lxstat64(_STAT_VER, path, stat_buf) shall behave as lstat64(path, stat_buf) as specified by Large File Support.

__fxstat64(_STAT_VER, fildes, stat_buf) shall behave as fstat64(fildes, stat_buf) as specified by Large File Support.

__xstat64(), __lxstat64(), and __fxstat64() are not in the source standard; they are only in the binary standard.

stat64(), lstat64(), and fstat64() are not in the binary standard; they are only in the source standard.

__environ

Name

__environ — alias for environ - user environment

Synopsis

extern char * *__environ;

Description

__environ is an alias for environ - user environment.

__nl_msg_cat_cntr

Name

__nl_msg_cat_cntr — new catalog load counter

Synopsis

#include <libintl.h>
extern int _nl_msg_cat_cntr;

**Description**

The global variable \_nl\_msg\_cat\_cntr is incremented each time a new catalog is loaded. This variable is only in the binary standard; it is not in the source standard.

**_sys\_errlist**

**Name**

\_sys\_errlist — array containing the "C" locale strings used by strerror()

**Synopsis**

```c
#include <stdio.h>
extern const char *const \_sys\_errlist[];
```

**Description**

\_sys\_errlist is an array containing the "C" locale strings used by strerror(). This normally should not be used directly. strerror() provides all of the needed functionality.

**_sys\_siglist**

**Name**

\_sys\_siglist — array containing the names of the signal names

**Synopsis**

```c
#include <signal.h>
extern const char *const \_sys\_siglist[NSIG];
```

**Description**

\_sys\_siglist is an array containing signal description strings ordered by signal number.

The \_sys\_siglist array is only in the binary standard; it is not in the source standard. Applications wishing to access signal descriptions should use the strsignal() function.
acct

Name
acct — switch process accounting on or off

Synopsis
#include <dirent.h>
int acct(const char * filename);

Description
When filename is the name of an existing file, acct() turns accounting on and appends a record to filename for each terminating process. When filename is NULL, acct() turns accounting off.

Return Value
On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

Errors
ENOSYS
BSD process accounting has not been enabled when the operating system kernel was compiled. The kernel configuration parameter controlling this feature is CONFIG_BSD_PROCESS_ACCT.

ENOMEM
Out of memory.

EPERM
The calling process has no permission to enable process accounting.

EACCES
filename is not a regular file.

EIO
Error writing to the filename.

EUSERS
There are no more free file structures or we run out of memory.
adjtime

Name
adjtime — correct the time to allow synchronization of the system clock

Synopsis
#include <time.h>
int adjtime(const struct timeval * delta, struct timeval * olddelta);

Description
adjtime() makes small adjustments to the system time as returned by gettimeofday(2), advancing or retarding it by the time specified by the timeval delta. If delta is negative, the clock is slowed down by incrementing it more slowly than normal until the correction is complete. If delta is positive, a larger increment than normal is used. The skew used to perform the correction is generally a fraction of one percent. Thus, the time is always a monotonically increasing function. A time correction from an earlier call to adjtime() may not be finished when adjtime() is called again. If olddelta is non-NULL, the structure pointed to will contain, upon return, the number of microseconds still to be corrected from the earlier call.

adjtime() may be used by time servers that synchronize the clocks of computers in a local area network. Such time servers would slow down the clocks of some machines and speed up the clocks of others to bring them to the average network time.

Appropriate privilege is required to adjust the system time.

Return Value
On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

Errors
EFAULT
An argument points outside the process's allocated address space.

EPERM
The process does not have appropriate privilege.
alphasort64

Name
alphasort64 — Comparison function for directory scanning (Large File Support)

Synopsis
#include <dirent.h>
int alphasort64(const struct dirent64 ** d1, const struct dirent64 ** d2);

Description
alphasort64() is a large-file version of the alphasort() function as defined in POSIX 1003.1-2008 (ISO/IEC 9945-2009). If differs only in that the d1 and d2 parameters are of type dirent64 instead of type dirent.
argz_add, argz_add_sep, argz_append, argz_count, argz_create, argz_create_sep, argz_delete, argz_extract, argz_insert, argz_next, argz_replace, argz_stringify

Name
argz_add, argz_add_sep, argz_append, argz_count, argz_create, argz_create_sep, argz_delete, argz_extract, argz_insert, argz_next, argz_replace, argz_stringify—Operate on argz vectors

Synopsis
#include <argz.h>
error_t argz_add(char ** argz, size_t * argz_len, const char * str);
error_t argz_add_sep(char ** argz, size_t * argz_len, const char * str, int sep);
error_t argz_append(char ** argz, size_t * argz_len, const char * buf, size_t buf_len);
size_t argz_count(const char * argz, size_t * argz_len);
error_t argz_create(char * const argv, char ** argz, size_t * argz_len);
error_t argz_create_sep(const char * str, int sep, char ** argz, size_t * argz_len);
void argz_delete(char ** argz, size_t * argz_len, char * entry);
void argz_extract(const char * argz, size_t argz_len, char ** argv);
error_t argz_insert(char ** argz, size_t * argz_len, char * before, const char * entry);
char argz_next(const char * argz, size_t argz_len, const char * entry);
error_t argz_replace(char ** argz, size_t * argz_len, const char * str, const char * with, unsigned int * replace_count);
void argz_stringify(char * argz, size_t argz_len, int sep);

Description
The argz functions operate on argz vectors, which are typically used to more easily manipulate program arguments, of the form described in ISO C (1999) in section 5.1.2.2.1, Program Startup. While an argv is an array of character pointers to strings, an argz vector is a set of strings, separated by null characters, in contiguous memory; the vector is described by a pointer to the first element and a size. There is no limitation that the argz must be made up of program arguments.

The argz functions which change argz vectors expect them to use memory allocated using malloc(), and will themselves use malloc() or realloc().

The argz_create() function converts an argv vector identified by argv to an argz vector with the same elements, identified by argz and argz_len.

The argz_create_sep() function converts the string identified by str, splitting into a separate string at each occurrence of sep, to an argz vector identified by argz and argz_len.

The argz_add() function adds the string identified by str to the vector identified by argz and argz_len, updating argz and argz_len.

The argz_add_sep() function adds the string identified by str, splitting into a separate string at each occurrence of sep, to the vector identified by argz, updating argz and argz_len.

The argz_append() function appends the argz vector identified by buf and buf_len to the argz vector identified by argz and argz_len, thus updating argz and argz_len.
The `argz_count()` function returns the number of strings in the `argz` vector identified by `argz` and `argz_len`.

The `argz_delete()` function removes the string identified by `entry` from the `argz` vector identified by `argz`, `argz_len`, updating `argz` and `argz_len`.

The `argz_extract()` function performs the inverse of `argz_create()`. It converts an `argz` vector identified by `argz` and `argz_len` to an `argv` vector identified by `argv` with the same elements.

The `argz_insert()` function inserts the string identified by `entry` at position before to the `argz` vector identified by `argz` and `argz_len`, updating `argz` and `argz_len`.

The `argz_next()` function returns the entry following the entry identified by `entry` in the `argz` vector identified by `argz` and `argz_len`. If `entry` is `NULL` the first entry is returned. This function can be used to step through an `argz` vector by obtaining the first entry by passing `NULL`, then passing the just obtained value to the next call, and so on. `NULL` is returned if there is no following entry.

The `argz_replace()` function replaces each occurrence of `str` in the `argz` vector identified by `argz` and `argz_len` with `with`, updating `argz` and `argz_len`. The counter pointed to by `replace_count` will be incremented by the number of replacements unless `NULL` is passed for `replace_count`.

The `argz_stringify()` function performs the inverse of `argz_create_sep()`. It converts the `argz` vector identified by `argz` and `argz_len` into a regular string, with the strings in the original vector separated by `sep` in the converted string. The conversion is done in place, so in effect each null byte in `argz` but the last one is replaced by `sep`.

**Return Value**

All of the `argz` functions that perform memory allocation return an `error_t` type. These functions return `0` on success; if memory allocation fails, they return `ENOMEM`.

`argz_count()` returns a count of substrings in the `argz` vector as a `size_t` type.

`argz_next()` returns a pointer to a substring in an `argz` vector, or `NULL`.

**See Also**

`envz_add`, `envz_entry`, `envz_get`, `envz_merge`, `envz_remove`, `envz_strip`
asprintf

Name

asprintf — write formatted output to a dynamically allocated string

Synopsis

```
#include <stdio.h>
int asprintf(char ** restrict ptr, const char * restrict format, ...);
```

Description

The `asprintf()` function shall behave as `sprintf()`, except that the output string shall be dynamically allocated space of sufficient length to hold the resulting string. The address of this dynamically allocated string shall be stored in the location referenced by `ptr`.

Return Value

Refer to `fprintf()`.

Errors

Refer to `fprintf()`.
backtrace, backtrace_symbols, backtrace_symbols_fd

Name
backtrace, backtrace_symbols, backtrace_symbols_fd — runtime stack back tracing

Synopsis
#include <execinfo.h>
int backtrace(void **array, int size);
char **backtrace_symbols(void *const *array, int size);
void backtrace_symbols_fd(void *const *array, int size, int fd);

description
backtrace() obtains a backtrace for the current thread as a list of pointers filled in to array. The size parameter describes the number of elements that will fit into array. backtrace() will truncate the list if necessary. A backtrace is a list of currently active function calls in a thread; each function call allocates a new stack frame and backtrace() obtains the return address from each stack frame.

backtrace_symbols() translates the information obtained from backtrace() into an array of strings. array is a pointer to an array of addresses as obtained from backtrace(). size is the number of entries in array, and should be the return value of the call to backtrace(). The strings contain the function name if it can be determined, a hexadecimal offset into the function, and the actual return address in hexadecimal. Note that the pointer returned by backtrace_symbols() is obtained by an internal call to malloc() and should be freed when no longer needed.

backtrace_symbols_fd() performs the same transformation as backtrace_symbols() given the same argument pair array, size, but writes the strings to the file descriptor contained in fd. This avoids the allocation of string space.

Return Value
backtrace() returns the number of entries placed into array, no more than size. If the value is less than size, the full backtrace was returned; else it may have been truncated.

On success, backtrace_symbols() returns a pointer to an array of strings, which will have size entries. On error, NULL is returned.

Errors
No errors are defined for these functions. If backtrace_symbols_fd() fails, it will be due to a failure in the call to malloc(), and errno will be set accordingly.

Notes
The ability to obtain useful backtrace information, in particular function names, is dependent on a number of factors at the time of program construction, such as compiler optimization options. Even if the program itself is constructed so as to make symbols visible, the call trace may descend into system libraries which have not been so constructed.

Inlined functions do not have stack frames, and functions declared as static are not exposed and so will not be available in the backtrace.

See Also
malloc()

basename

Name
basename — return the last component of a file name

Synopsis

#include <libgen.h>
char * basename(const char * path);

Description

In the source standard, basename() is implemented as a macro causing it to behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), and is equivalent to the function __xpg_basename(). If the macro is undefined, basename() from the binary standard is used, with differences as described here:

The string identified by path shall not be modified.

If path is "/", or ends with a trailing '//' character, the basename() function shall return a pointer to an empty string.

Return Value

On success, the basename() function shall return a pointer to the final component of path. Otherwise, it shall return a null pointer.

See Also

__xpg_basename()}
bind_textdomain_codeset

Name
bind_textdomain_codeset — specify encoding for message retrieval

Synopsis
#include <libintl.h>
char * bind_textdomain_codeset (const char * domainname, const char * codeset);

Description
The bind_textdomain_codeset() function can be used to specify the output codeset for message catalogs for domain domainname. The codeset argument shall be a valid codeset name which can be used for the iconv_open function, or a null pointer. If the codeset argument is the null pointer, then function returns the currently selected codeset for the domain with the name domainname. It shall return a null pointer if no codeset has yet been selected.

Each successive call to bind_textdomain_codeset() function overrides the settings made by the preceding call with the same domainname.

The bind_textdomain_codeset() function shall return a pointer to a string containing the name of the selected codeset. The string shall be allocated internally in the function and shall not be changed or freed by the user.

Parameters
domainname
The domainname argument is applied to the currently active LC_MESSAGES locale. It is equivalent in syntax and meaning to the domainname argument to textdomain, except that the selection of the domain is valid only for the duration of the call.

codeset
The name of the output codeset for the selected domain, or NULL to select the current codeset.

If domainname is the null pointer, or is an empty string, bind_textdomain_codeset() shall fail, but need not set errno.

Return Value
Returns the currently selected codeset name. It returns a null pointer if no codeset has yet been selected.

Errors
ENOMEM
Insufficient memory available to allocate return value.

See Also
ggettext, dgettext, ngettext, dnggettext, dcgettext, dcngettext, textdomain, bindtextdomain
bindresvport

Name
bindresvport — bind socket to privileged IP port

Synopsis
#include <sys/types.h>
#include <rpc/rpc.h>
int bindresvport(int sd, struct sockaddr_in * sin);

Description
If the process has appropriate privilege, the bindresvport() function shall bind a
socket to an anonymous privileged IP port, that is, arbitrarily selected from the range
512 through 1023.

If the bind is successful and sin is not NULL, and the port number bound to is returned
in the sin_port member of sin. Any caller-supplied value of sin_port is ignored.

If sin is NULL, the address family is taken to be AF_INET and an available privileged
port is bound to. Since there is no sockaddr_in structure, the port number chosen cannot
be returned. The getsockname() may be used to query for this information.

Return Value
On success, 0 is returned. On error, -1 is returned and errno is set to indicate the error.

Errors
bindresvport() may fail in the same way as bind() in POSIX_1003.1-2008
(ISO/IEC 9945-2009). The following additional or differing failures may occur:

EADDRINUSE
All privileged ports are in use.

EAFNOSUPPORT
The specified address is not a valid address for the address family of the specified
socket, or the address family is not supported.

EPFNOSUPPORT
The same meaning as EAFNOSUPPORT. Some older implementations may return
this error instead.

Note: At this time, only AF_INET is supported. Applications should be prepared for either
the EAFNOSUPPORT or EPFNOSUPPORT error to be indicated.
bindtextdomain

Name
bindtextdomain — specify the location of a message catalog

Synopsis
#include <libintl.h>
char * bindtextdomain(const char * domainname, const char * dirname);

Description
The bindtextdomain() shall set the the base directory of the hierarchy containing message catalogs for a given message domain.

The bindtextdomain() function specifies that the domainname message catalog can be found in the dirname directory hierarchy, rather than in the system default locale data base.

If dirname is not NULL, the base directory for message catalogs belonging to domain domainname shall be set to dirname. If dirname is NULL, the base directory for message catalogs shall not be altered.

The function shall make copies of the argument strings as needed.

dirname can be an absolute or relative pathname.

Note: Applications that wish to use chdir() should always use absolute pathnames to avoid misadventently selecting the wrong or non-existant directory.

If domainname is the null pointer, or is an empty string, bindtextdomain() shall fail, but need not set errno.

The bindtextdomain() function shall return a pointer to a string containing the name of the selected directory. The string shall be allocated internally in the function and shall not be changed or freed by the user.

Return Value
On success, bindtextdomain() shall return a pointer to a string containing the directory pathname currently bound to the domain. On failure, a NULL pointer is returned, and the global variable errno may be set to indicate the error.

Errors
ENOMEM
Insufficient memory was available.

See Also
gettext, dgettext, ngettext, dnlgettext, dcgettext, dcngettext, textdomain, bind_textdomain_codeset
**cfmakeraw**

**Name**

*cfmakeraw* — get and set terminal attributes

**Synopsis**

```c
#include <termios.h>
void cfmakeraw(struct termios * termios_p);
```

**Description**

The *cfmakeraw()* function shall set the attributes of the *termios* structure referenced by *termios_p* as follows:

```c
    termios_p->c_iflag &= ~(IGNBRK|BRKINT|PARMRK|ISTRIP
                           |INLCR|IGNCR|ICRNL|IXON);
    termios_p->c_oflag &= ~OPOST;
    termios_p->c_lflag &= ~(ECHO|ECHONL|ICANON|ISIG|IEXTEN);
    termios_p->c_cflag &= ~(CSIZE|PARENB);
    termios_p->c_cflag |= CS8;
```

*termios_p* shall point to a *termios* structure that contains the following members:

```c
tcflag_t c_iflag;      /* input modes */
tcflag_t c_oflag;      /* output modes */
tcflag_t c_cflag;      /* control modes */
tcflag_t c_lflag;      /* local modes */
cc_t c_cc[NCCS];       /* control chars */
```
**cfsetspeed**

**Name**
cfsetspeed — set terminal input and output data rate

**Synopsis**
```
#include <termios.h>
int cfsetspeed(struct termios *t, speed_t speed);
```

**Description**
The cfsetspeed() function shall set the input and output speeds in t to the value specified by speed. The effects of the function on the terminal as described below do not become effective, nor are all errors detected, until the tcsetattr() function is called. Certain values for baud rates set in termios and passed to tcsetattr() have special meanings.

**Return Value**
On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

**Errors**
EINVAL
Invalid speed argument

**clearerr_unlocked**

**Name**
clearerr_unlocked — non-thread-safe clearerr

**Description**
clearerr_unlocked() is the same as clearerr(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().
daemon

Name
d daemon — run in the background

Synopsis
#include <unistd.h>
int daemon(int nochdir, int noclose);

Description
The daemon() function shall create a new process, detached from the controlling terminal. If successful, the calling process shall exit and the new process shall continue to execute the application in the background. If nochdir evaluates to true, the current directory shall not be changed. Otherwise, daemon() shall change the current working directory to the root (/). If noclose evaluates to true the standard input, standard output, and standard error file descriptors shall not be altered. Otherwise, daemon() shall close the standard input, standard output and standard error file descriptors and reopen them attached to /dev/null.

Return Value
On error, -1 is returned, and the global variable errno is set to any of the errors specified for the library functions fork() and setsid().

dcgettext

Name
dcgettext — perform domain and category specific lookup in message catalog

Synopsis
#include <libintl.h>
```c
#include <locale.h>
char * dcgettext(const char * domainname, const char * msgid, int category);
```

**Description**

The `dcgettext()` function is a domain specified version of `gettext()`.

The `dcgettext()` function shall lookup the translation in the current locale of the message identified by `msgid` in the domain specified by `domainname` and in the locale category specified by `category`. If `domainname` is NULL, the current default domain shall be used. The `msgid` argument shall be a NULL-terminated string to be matched in the catalogue. `category` shall specify the locale category to be used for retrieving message strings. The category parameter shall be one of `LC_CTYPE`, `LC_COLLATE`, `LC_MESSAGES`, `LC_MONETARY`, `LC_NUMERIC`, or `LC_TIME`. The default domain shall not be changed by a call to `dcgettext()`.

**Return Value**

If a translation was found in one of the specified catalogs, it shall be converted to the current locale's codeset and returned. The resulting NULL-terminated string shall be allocated by the `dcgettext` function, and must not be modified or freed. If no translation was found, or category was invalid, `msgid` shall be returned.

**Errors**

`dcgettext()` shall not modify the `errno` global variable.

**See Also**

`gettext`, `dgettext`, `ngettext`, `dngettext`, `dcngettext`, `textdomain`, `bindtextdomain`, `bind_textdomain_codeset`

**dcngettext**

**Name**

dcngettext — perform domain and category specific lookup in message catalog with plural

**Synopsis**

```c
#include <libintl.h>
```
#include <locale.h>
char * dcngettext(const char * domainname, const char * msgid1, const char * msgid2, unsigned long int n, int category);

Description
The dcngettext() function is a domain specific version of gettext, capable of returning either a singular or plural form of the message. The dcngettext() function shall lookup the translation in the current locale of the message identified by msgid1 in the domain specified by domainname and in the locale category specified by category. If domainname is NULL, the current default domain shall be used. The msgid1 argument shall be a NULL-terminated string to be matched in the catalogue. category shall specify the locale category to be used for retrieving message strings. The category parameter shall be one of LC_CTYPE, LC_COLLATE, LC_MESSAGES, LC_MONETARY, LC_NUMERIC, or LC_TIME. The default domain shall not be changed by a call to dcngettext(). If n is 1 then the default domain is returned, otherwise one of the plural forms is returned, depending on the value of n and the current locale settings.

Return Value
If a translation corresponding to the value of n was found in one of the specified catalogs for msgid1, it shall be converted to the current locale's codeset and returned. The resulting NULL-terminated string shall be allocated by the dcngettext() function, and must not be modified or freed. If no translation was found, or category was invalid, msgid1 shall be returned if n has the value 1, otherwise msgid2 shall be returned.

Errors
dcngettext() shall not modify the errno global variable.

See Also
ggettext, dggettext, ngettext, dngettext, dgettext, dcgettext, textdomain, bindtextdomain, bind_textdomain_codeset
dgettext

Name

dgettext — perform lookup in message catalog for the current LC_MESSAGES locale

Synopsis

#include <libintl.h>
char * dgettext(const char * domainname, const char * msgid);

Description

dgettext() is a domain specified version of gettext().

The dgettext() function shall search the currently selected message catalogs in the domain domainname for a string identified by the string msgid. If a string is located, that string shall be returned. The domain specified by domainname applies to the currently active LC_MESSAGE locale. The default domain shall not be changed by a call to dgettext().

Note: The usage of domainname is equivalent in syntax and meaning to the textdomain() function's application of domainname, except that the selection of the domain in dgettext() is valid only for the duration of the call.

The dgettext() function is equivalent to dcgettext(domainname, msgid, LC_MESSAGES).

Return Value

On success of a msgid query, the translated NULL-terminated string is returned. On error, the original msgid is returned. The length of the string returned is undetermined until dgettext() is called.

Errors

dgettext() shall not modify the errno global variable.

See Also

gettext, dgettext, ngettext, dngettext, dcgettext, dcngettext, textdomain, bindtextdomain, bind_textdomain_codeset
**dl_iterate_phdr**

**Name**

dl_iterate_phdr — iterate over a program's loaded shared objects

**Synopsis**

```c
#include <link.h>
int dl_iterate_phdr(int(*callback) (struct dl_phdr_info *, size_t, void *), void *data);
```

**Description**

dl_iterate_phdr() allows a program to iterate over the shared objects it has loaded. The function described by the callback parameter is called once for each loaded shared object, allowing an action to be taken for each one. callback is called with three arguments which are filled in by the implementation: a pointer to a structure of type dl_phdr_info containing information about the shared object; an integer size of the structure; and a copy of the data argument to dl_iterate_phdr(). If callback returns a non-zero value, dl_iterate_phdr() will stop processing, even if there are unprocessed shared objects. The order of processing is unspecified.

The dl_phdr_info structure has the following members (note that on 64-bit architectures the types here shown as Elf32_type will instead be Elf64_type):

- `Elf32_Addr dlpi_addr;`
- `const char *dlpi_name;`
- `const Elf32_Phdr *dlpi_phdr;`
- `Elf32_Half dlpi_phnum;`
- `unsigned long long int dlpi_adds;`
- `unsigned long long int dlpi_subs;`
- `size_t dlpi_tls_modid;`
void *dlpi_tls_data;

dlpi_addr contains the base address of the shared object.

dlpi_name is a null-terminated string giving the pathname from which the shared object was loaded.

dlpi_phdr is a pointer to an array of program headers for this shared object, while dlpi_phnum is the number of entries in this array.

dlpi_adds and dlpi_subs are incremented when shared objects are added or removed, respectively.

dlpi_tls_modid contains the module ID used in TLS relocations, if there is a PT_TLS segment. Otherwise the value shall be zero.

dlpi_tls_data contains the address of the calling thread's instance of this module's PT_TLS segment, if there is one and it has been allocated in the calling thread. Otherwise the value shall be a null pointer.

Some implementations may not provide all fields in dl_phdr_info, although the first four are always mandatory. Applications are advised to have the callback function check the size parameter before examining the later members.

**Return Value**

The dl_iterate_phdr() function returns whatever value was returned by the last call to callback. This will be zero if processing completed normally, since processing does not continue unless the callback function returns zero.

**Errors**

No errors are defined by dl_iterate_phdr(); as noted the callback function must use a zero return to indicate success but may assign any meaning it wishes to non-zero returns.

**dngettext**

**Name**

dngettext — perform lookup in message catalog for the current locale

**Synopsis**

```c
#include <libintl.h>

char * dngettext(const char * domainname, const char * msgid1, const char * msgid2, unsigned long int n);
```

**Description**

dngettext() shall be equivalent to a call to
dcngettext(domainname, msgid1, msgid2, n, LC_MESSAGES)

See dcngettext() for more information.

**See Also**

ggettext, dggettext, ngettext, dcgettext, dcngettext, textdomain, bindtextdomain, bind_textdomain_codeset
drand48_r

Name
drand48_r — reentrantly generate pseudorandom numbers in a uniform distribution

Synopsis
#include <stdlib.h>
int drand48_r(struct drand48_data * buffer, double * result);

Description
The interface drand48_r() shall function in the same way as the interface
drand48(), except that drand48_r() shall use the data in buffer instead of the
global random number generator state.
Before it is used, buffer must be initialized, for example, by calling lcong48_r(),
seed48_r(), or srand48_r(), or by filling it with zeroes.

dendutent

Name
dendutent — access utmp file entries

Synopsis
#include <utmp.h>
void endutent(void);

Description
dendutent() closes the utmp file. It should be called when the user code is done ac-
cessing the file with the other functions.
envz_add, envz_entry, envz_get, envz_merge, envz_remove, envz_strip

Name
envz_add, envz_entry, envz_get, envz_merge, envz_remove, envz_strip — Operate on environment vectors

Synopsis
#include <envz.h>
error_t envz_add(char ** envz, size_t * envz_len, const char * name, const char * value);
char envz_entry(const char * envz, size_t envz_len, const char * name);
char envz_get(const char * envz, size_t envz_len, const char * name);
error_t envz_merge(char ** envz, size_t * envz_len, const char * envz2, size_t envz2_len, int override);
void envz_remove(char ** envz, size_t * envz_len, const char * name);
void envz_strip(char ** envz, size_t * envz_len);

Description
The envz functions operate on envz vectors, which are typically used to manipulate program environment variables.

An envz vector is identical in makeup to an argz vector (see argz_add, argz_add_sep, argz_append, argz_count, argz_create, argz_create_sep, argz_delete, argz_extract, argz_insert, argz_next, argz_replace, argz_stringify) but has the constraint that each element is a name, value pair separated by an = character. Only the first = character in an element has special meaning, any subsequent instances are part of the value string. If no = character is present in an element, the value is taken to be NULL. If an element has an empty value (an = character is present), the value will return the empty string "" when queried.

Since an envz vector is an argz vector, the argz functions can be used where it makes sense. For example, converting from a program's environment variables (as described in Chapter 8 of the XBD volume of POSIX 1003.1-2008 (ISO/IEC 9945-2009)) to an envz vector is done with argz_create().

The envz_add() function adds a string constructed from name and value in the form "name=value" to the envz vector identified by envz and envz_len, updating envz and envz_len. If value is NULL it adds a string of the form "name". If an entry with the same name already exists, it is replaced.

The envz_entry() function searches for name in the envz vector identified by envz and envz_len, returning the full entry if found, or NULL if not.

The envz_get() function searches for name in the envz vector identified by envz and envz_len, returning the value part of the entry if found, or NULL if not. Note the value may be also NULL.

The envz_merge() function adds each entry from the envz vector identified by envz2 and envz2_len to the envz vector identified by envz and envz_len, updating envz and envz_len. The behavior is as if envz_add() were called for each entry in envz2. If override is true, then values from envz2 will replace those with the same name in envz.

The envz_remove() function removes the entry for name from the envz vector identified by envz and envz_len if it exists, updating envz and envz_len.
The `envz_strip()` function removes all entries with value `NULL`.

**Return Value**

The `envz` functions that perform memory allocation (`envz_add()` and `envz_merge()`) return an `error_t` type. These functions return `0` on success; if memory allocation fails, they return `ENOMEM`.

`envz_entry()` and `envz_get()` return a pointer to a substring in an `envz` vector, or `NULL`.

**See Also**

`argz add`, `argz add sep`, `argz append`, `argz count`, `argz create`, `argz create sep`, `argz delete`, `argz extract`, `argz insert`, `argz next`, `argz replace`, `argz stringify`
epoll_create

Name
epoll_create — open an epoll file descriptor

Synopsis
#include <sys/epoll.h>
int epoll_create(int size);

Description
The epoll API, which consists of the interfaces epoll_create(), epoll_ctl(), and epoll_wait(), shall support all file descriptors compatible with poll(). These interfaces shall be usable in either level-triggered or edge-triggered mode. In level-triggered mode, epoll has similar semantics to poll(), and can be used as a faster replacement for it. In edge-triggered mode, epoll shall only report events for a file descriptor when changes occur on it.

The epoll_create() interface shall open an epoll file descriptor by allocating an event backing store of approximately size size. The size parameter is a hint to the kernel about how large the event storage should be, not a rigidly-defined maximum size.

Return Value
On success, epoll_create() shall return the file descriptor, a non-negative integer that shall be used for subsequent epoll calls. It should be closed with the close() function.

On failure, epoll_create() shall return -1 and set errno as follows.

Errors
EINVAL
The size parameter is not positive.
ENFILE
The maximum number of open files has been reached by the system.
ENOMEM
Not enough memory to create the kernel object.

See Also
close(), epoll_ctl(), epoll_wait(), poll.
epoll_ctl

Name
epoll_ctl — control an epoll file descriptor

Synopsis
#include <sys/epoll.h>
int epoll_ctl(int epfd, int op, int fd, struct epoll_event * event);

Description
The interface epoll_ctl() shall control an epoll file descriptor.

The parameter epfd shall specify the epoll file descriptor to control.
The parameter op shall specify the operation to perform on the specified target file
descriptor.
The parameter fd shall specify the target file descriptor on which to perform the spec-
cified operation.
The parameter event shall specify the object associated with the target file descriptor.
The events member of the event parameter is a bit set composed of the event types
listed below.

Event types

EPOLLERR
An error condition occurred on the target file descriptor. It shall not be necessary to
set this event in events; this interface shall always wait for it.

EPOLLET
This event shall set edge-triggered behavior for the target file descriptor. The
default epoll behavior shall be level-triggered.

EPOLLHUP
A hang up occurred on the target file descriptor. It shall not be necessary to set this
event in events; this interface shall always wait for it.

EPOLLIN
The file is accessible to read() operations.

EPOLLONESHOT
This event shall set one-shot behavior for the target file descriptor. After
epoll_wait() retrieves an event, the file descriptor shall be disabled and epoll
shall not report any other events. To reenable the file descriptor with a new event
mask, the user should invoke epoll_ctl() with EPOLL_CTL_MOD in the op
parameter.

EPOLLOUT
The file is accessible to write() operations.

EPOLLPRI
Urgent data exists for read() operations.

EPOLLRDHUP
A stream socket peer closed the connection, or else the peer shut down the writing half of the connection.

**Values of the op parameter**

EPOLL_CTL_ADD

Associate event with the file described by fd, and add fd to the epoll descriptor epfd.

EPOLL_CTL_DEL

Remove fd from epfd, and ignore event, which can be NULL.

EPOLL_CTL_MOD

Change the event event associated with fd.

**Return Value**

On success, epoll_ctl() shall return 0.

On failure, epoll_ctl() shall return -1 and set errno as follows.

**Errors**

EBADF

The parameter epfd or the parameter fd is an invalid file descriptor.

EEXIST

The parameter op was EPOLL_CTL_ADD, but the file descriptor fd is already in epfd.

EINVAL

The parameter epfd is invalid, or it is the same as fd, or the operation specified by the parameter op is unsupported.

ENOENT

The parameter op was EPOLL_CTL_MOD or EPOLL_CTL_DEL, but the file descriptor fd is not in epfd.

ENOMEM

Not enough memory for the operation specified by the parameter op.

EPERM

The file specified by fd does not support epoll.

**See Also**

close(), epoll_create(), epoll_wait(), poll().
epoll_wait

Name

epoll_wait — wait for I/O events on an epoll file descriptor

Synopsis

#include <sys/epoll.h>
int epoll_wait(int epfd, struct epoll_event * events, int maxevents, int timeout);

Description

The interface epoll_wait() shall wait for events on the epoll file descriptor specified by the parameter epfd.

Upon success, the output parameter events shall refer to an area of memory containing epoll_event structures available to the caller. The data members of these structures shall contain the data set by the user with the interface epoll_ctl(). The events members shall contain the event bit field that was returned.

The parameter maxevents shall specify the maximum number of events that epoll_wait() may return in the output parameter events. The value of this parameter should be greater than 0.

The parameter timeout shall specify the maximum number of milliseconds that epoll_wait() shall wait for events. If the value of this parameter is 0, then epoll_wait() shall return immediately, even if no events are available, in which case the return code shall be 0. If the value of timeout is -1, then epoll_wait() shall block until either a requested event occurs or the call is interrupted.

Return Value

On success, epoll_wait() shall return the number of file descriptors that are ready for the I/O that was requested, or else 0 if no descriptors became ready during timeout.

On failure, epoll_wait() shall return -1 and set errno as follows.

Errors

EBADF

The parameter epfd is not a valid file descriptor.

EFAULT

The area of memory referenced by the parameter events cannot be accessed with write permissions.

EINTR

The call was interrupted by a signal handler before the timeout expired or any requested event took place.

EINVAL

The parameter epfd is not a valid epoll file descriptor, or else the parameter maxevents is less than or equal to 0.

See Also

close(), epoll_ctl(), epoll_create(), poll().
**erand48_r**

**Name**
erand48_r — reentrantly generate pseudorandom numbers in a uniform distribution

**Synopsis**

```c
#include <stdlib.h>
int erand48_r(unsigned short[3] xsubi, struct drand48_data * buffer, double * result);
```

**Description**
The interface `erand48_r()` shall function in the same way as the interface `erand48()`, except that `erand48_r()` shall use the data in `buffer` instead of the global random number generator state.

Before it is used, `buffer` must be initialized, for example, by calling `lcong48_r()`, `seed48_r()`, or `srand48_r()`, or by filling it with zeroes.

**err**

**Name**
err — display formatted error messages

**Synopsis**

```c
#include <err.h>
void err(int eval, const char * fmt, ...);
```

**Description**
The `err()` function shall display a formatted error message on the standard error stream. First, `err()` shall write the last component of the program name, a colon character, and a space character. If `fmt` is non-NULL, it shall be used as a format string for the `printf()` family of functions, and `err()` shall write the formatted message, a colon character, and a space. Finally, the error message string affiliated with the current value of the global variable `errno` shall be written, followed by a newline character.

The `err()` function shall not return, the program shall terminate with the exit value of `eval`.

**See Also**
error(), errx()

**Return Value**
None.

**Errors**
None.
error

Name
error — print error message

Synopsis
#include <error.h>
void error (int status , int errnum , const char * format , ...);

Description
error() shall print a message to standard error.

error() shall build the message from the following elements in their specified order:

1. the program name. If the application has provided a function named
   error_print_progname(), error() shall call this to supply the program
   name; otherwise, error() uses the content of the global variable
   program_name.

2. the colon and space characters, then the result of using the printf-style Format
   and the optional arguments.

3. if errnum is nonzero, error() shall add the colon and space characters, then the
   result of strerror(errnum).

4. a newline.

If status is nonzero, error() shall call exit(status).

See Also
err(), errx()
errx

Name
errx — display formatted error message and exit

Synopsis
#include <err.h>
void errx (int eval, const char * fmt, ...);

Description
The errx() function shall display a formatted error message on the standard error stream. The last component of the program name, a colon character, and a space shall be output. If fmt is non-NULL, it shall be used as the format string for the printf() family of functions, and the formatted error message, a colon character, and a space shall be output. The output shall be followed by a newline character.

errx() does not return, but shall exit with the value of eval.

Return Value
None.

Errors
None.

See Also
error(), err()
fcntl

Name
fcntl — file control

Description
fcntl() is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

Implementation may set O_LARGEFILE

According to POSIX 1003.1-2008 (ISO/IEC 9945-2009), only an application sets fcntl() flags, for example O_LARGEFILE. However, this specification also allows an implementation to set the O_LARGEFILE flag in the case where the programming environment is one of _POSIX_V6_ILP32_OFFBIG, _POSIX_V6_LP64_OFF64, _POSIX_V6_LPBIG_OFFBIG. See getconf and c99 in POSIX 1003.1-2008 (ISO/IEC 9945-2009) for a description of these environments. Thus, calling fcntl() with the F_GETFL command may return O_LARGEFILE as well as flags explicitly set by the application in the case that both the implementation and the application support an off_t of at least 64 bits.

Additional flags

In addition to the available values for cmd, as documented in POSIX 1003.1-2008 (ISO/IEC 9945-2009), this specification permits the following constants.

F_GETSIG shall get the number of the signal to be sent when input or output can occur. If the value is 0, then SIGIO shall be sent. Otherwise, the value retrieved shall be the signal sent, and the signal handler can discover more information when installed with the SA_SIGINFO flag.

F_SETSIG shall set the number of the signal to be sent when input or output can occur. If the value is 0, then SIGIO shall be sent. Otherwise, the value set shall be the signal to be sent, and the signal handler can discover more information when installed with the SA_SIGINFO flag.

F_GETLK64 is analogous to the F_GETLK constant in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but shall provide a 64-bit interface on non-64-bit architectures. It is identical to F_GETLK on a 64-bit machine, but is provided in 64-bit environments for source code consistency among architectures.

F_SETLK64 is analogous to the F_SETLK constant in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but shall provide a 64-bit interface on non-64-bit architectures. It is identical to F_SETLK on a 64-bit machine, but is provided in 64-bit environments for source code consistency among architectures.

F_SETLKW64 is analogous to the F_SETLKW constant in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but provides a 64-bit interface on non-64-bit architectures. It is identical to F_SETLKW on a 64-bit machine, but is provided in 64-bit environments for source code consistency among architectures.
feof_unlocked

Name
feof_unlocked — non-thread-safe feof

Description
feof_unlocked() is the same as feof(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().

ferror_unlocked

Name
ferror_unlocked — non-thread-safe ferror

Description
ferror_unlocked() is the same as ferror(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().

fflush_unlocked

Name
fflush_unlocked — non-thread-safe fflush

Description
fflush_unlocked() is the same as fflush() except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().

fgetc_unlocked

Name
fgetc_unlocked — non-thread-safe fgetc

Description
fgetc_unlocked() is the same as fgetc(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().

fgets_unlocked

Name
fgets_unlocked — non-thread-safe fgets

Description
fgets_unlocked() is the same as fgets(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().
fgetwc_unlocked

Name
fgetwc_unlocked — non thread safe fgetwc

Description
fgetwc_unlocked() is the same as fgetwc() except that it need not be thread safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().

fgetws_unlocked

Name
fgetws_unlocked — non-thread-safe fgetws

Description
fgetws_unlocked() is the same as fgetws(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().

fileno_unlocked

Name
fileno_unlocked — non-thread-safe fileno

Description
fileno_unlocked() is the same as fileno(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().
flock

Name
flock — apply or remove an advisory lock on an open file

Synopsis
int flock(int fd, int operation);

Description
flock() applies or removes an advisory lock on the open file fd. Valid operation types are:

LOCK_SH
   Shared lock. More than one process may hold a shared lock for a given file at a given time.

LOCK_EX
   Exclusive lock. Only one process may hold an exclusive lock for a given file at a given time.

LOCK_UN
   Unlock.

LOCK_NB
   Don't block when locking. May be specified (by oring) along with one of the other operations.

A single file may not simultaneously have both shared and exclusive locks.

Return Value
On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

Errors
EWOULDBLOCK
   The file is locked and the LOCK_NB flag was selected.

EBADF
   fd is not a not an open file descriptor.

EINTR
   While waiting to acquire a lock, the call was interrupted by delivery of a signal caught by a handler.

EINVAL
   The operation is invalid.

ENOLCK
   The implementation ran out of memory for allocating lock records.
fnmatch

**Name**
fnmatch — match a filename or a pathname

**Description**
fnmatch() is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

**Additional flags**
In addition to the available values that can be used to form flags, as documented in POSIX 1003.1-2008 (ISO/IEC 9945-2009), this specification permits the following constants.

**FNM_CASEFOLD**
If this flag is set, the pattern is matched case-insensitively.

**FNM_FILE_NAME**

fputc_unlocked

**Name**
fputc_unlocked — non-thread-safe fputc

**Description**
fputc_unlocked() is the same as fputc(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().

fputs_unlocked

**Name**
fputs_unlocked — non-thread-safe fputs

**Description**
fputs_unlocked() is the same as fputs(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().

fputwc_unlocked

**Name**
fputwc_unlocked — non-thread-safe fputwc

**Description**
fputwc_unlocked() is the same as fputwc(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().
**fputws_unlocked**

**Name**

fputws_unlocked — non-thread-safe fputws

**Description**

fputws_unlocked() is the same as fputws(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().

**fread_unlocked**

**Name**

fread_unlocked — non-thread-safe fread

**Description**

fread_unlocked() is the same as fread(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().

**fscanf**

**Name**

fscanf — convert formatted input

**Description**

The scanf() family of functions shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except as noted below.

**Differences**

The %s, %S and %[^ conversion specifiers shall accept an option length modifier a, which shall cause a memory buffer to be allocated to hold the string converted. In such a case, the argument corresponding to the conversion specifier should be a reference to a pointer value that will receive a pointer to the allocated buffer. If there is insufficient memory to allocate a buffer, the function may set error to ENOMEM and a conversion error results.

**Note:** This directly conflicts with the ISO C (1999) usage of %a as a conversion specifier for hexadecimal float values. While this conversion specifier should be supported, a format specifier such as "%aseconds" will have a different meaning on an LSB conforming system.
fstatfs

Name
fstatfs — (deprecated)

Synopsis
#include <sys/statfs.h>
int fstatfs(int fd, struct statfs *buf);

Description
The fstatfs() function returns information about a mounted file system. The file system is identified by fd, a file descriptor of an open file within the mounted filesystem. The results are placed in the structure pointed to by buf.

Fields that are undefined for a particular file system shall be set to 0.

Note: Application developers should use the fstatvfs() function to obtain general file system information. Applications should only use the fstatfs() function if they must determine the file system type, which need not be provided by fstatvfs().

Return Value
On success, the fstatfs() function shall return 0 and set the fields of the structure identified by buf accordingly. On error, the fstatfs() function shall return -1 and set errno accordingly.

Errors
EBADF
fd is not a valid open file descriptor.

EFAULT
buf points to an invalid address.

EIO
An I/O error occurred while reading from or writing to the file system.

ENOSYS
The filesystem fd is open on does not support statfs().
**fstatfs64**

**Name**

fstatfs64 — (deprecated)

**Synopsis**

```c
#include <sys/statfs.h>
int fstatfs64(int fd, struct statfs64 *buf);
```

**Description**

The `fstatfs64()` function returns information about a mounted file system. The file system is identified by `fd`, a file descriptor of an open file within the mounted file system. The results are placed in the structure pointed to by `buf`.

Fields that are undefined for a particular file system shall be set to 0.

`fstatfs64()` is a large-file version of the `fstatfs()` function.

**Note:** Application developers should use the `fstatvfs64()` function to obtain general file system information. Applications should only use the `fstatfs64()` function if they must determine the file system type, which need not be provided by `fstatvfs64()`.

**Return Value**

On success, the `fstatfs64()` function shall return 0 and set the fields of the structure identified by `buf` accordingly. On error, the `fstatfs64()` function shall return -1 and set `errno` accordingly.

**Errors**

See `fstatfs()`.
futimes

Name
futimes, lutimes — set file access and modification times

Synopsis
#include <sys/time.h>
int futimes(int fd, const struct timeval tv[2]);
int lutimes(const char *filename, const struct timeval tv[2]);

Description
The futimes() and lutimes() functions shall set the access and modification times of a file to the values of the tv argument, which is an array of two timeval structures. The behavior is as for utimes() in POSIX 1003.1-2008 (ISO/IEC 9945-2009).

The futimes() function shall change the times of of the open file described by file descriptor fd.

The lutimes() function shall change the times of of the file pointed to by the filename argument, except that if filename refers to a symbolic link, then the link is not followed and the times of the symbolic link are changed. This is similar to supplying AT_SYMLINK_NOFOLLOW in the flag argument to the utimensat() function.

Errors
As for utimes(), but in addition:

ENOSYS
  This implementation does not support this function (for lutimes()).
  The implementation could not access a resource needed to complete the function (for futimes()).

See Also
utimes(), utime(), utimensat().

fwrite_unlocked

Name
fwrite_unlocked — non-thread-safe fwrite

Description
fwrite_unlocked() is the same as fwrite(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().
fwscanf

Name
fwscanf — convert formatted input

Description
The scanf() family of functions shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except as noted below.

Differences
The %s, %S and %[@ conversion specifiers shall accept an option length modifier a, which shall cause a memory buffer to be allocated to hold the string converted. In such a case, the argument corresponding to the conversion specifier should be a reference to a pointer value that will receive a pointer to the allocated buffer. If there is insufficient memory to allocate a buffer, the function may set errno to ENOMEM and a conversion error results.

Note: This directly conflicts with the ISO C (1999) usage of %a as a conversion specifier for hexadecimal float values. While this conversion specifier should be supported, a format specifier such as “%aseconds” will have a different meaning on an LSB conforming system.

getcwd

Name
getcwd — get the pathname of the current working directory

Synopsis
#include <unistd.h>
char * getcwd(char * buf, size_t size);

Description
The getcwd() functions shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except as noted below.

Differences
If buf is NULL, memory is allocated for buf. If size is 0, the allocation size will be the size of the pathname +1, else the requested size is allocated.

Changed or Added Errors
EINVAL
  The size argument is 0 and buf is not a null pointer.
ENOENT
  The current working directory has been unlinked.
**getdomainname**

**Name**

getdomainname — get NIS domain name (DEPRECATED).

**Synopsis**

```
#include <unistd.h>
int getdomainname (char * name, size_t namelen);
```

**Description**

If the Network Information System (NIS) is in use, getdomainname() shall copy the NIS domain name to the supplied buffer identified by *name*, with maximum length *namelen*. If the NIS domain name is not currently set, getdomainname() shall copy the string "(none)" to the *name*. If *namelen* is less than the length of the string to be copied, getdomainname() shall either truncate the string to *namelen* characters and place it in *name* (without a terminating null character), or shall fail with EINVAL.

**Note:** The NIS domain name is not the same as the domain portion of a fully qualified domain name (for example, in DNS).

The LSB does not include other NIS functions, nor does it specify how NIS may affect other database functions. No conforming application can make use of this information beyond noting whether or not the domain name has been set. If the name is set to a value other than the string "(none)", the application should not imply that NIS is in use. Similarly, if it is set to "(none)", the application should not assume that NIS is not in use, although NIS functionality may be restricted in this case.

**Return Value**

On success, getdomainname() shall return 0. Otherwise, it shall return -1 and set *errno* to indicate the error.

**Errors**

EINVAL

*name* is a null pointer.

EINVAL

The buffer identified by *name* and *namelen* is of insufficient size to store the NIS domain name string, and the implementation considers this an error.

**Future Directions**

The LSB does not include other NIS interfaces, and a future version of this specification may remove this interface. Application developers should avoid using this interface where possible.
getdtablesize

Name
getdtablesize — get file descriptor table size (DEPRECATED)

Synopsis
#include <unistd.h>
int getdtablesize (void);

Description
The function getdtablesize() returns the number of files a process can have open.

Note: The getdtablesize() function is deprecated. Portable applications should call sysconf() with the _SC_OPEN_MAX option instead.

Return Value
The getdtablesize() function returns the current soft limit as if obtained by a call to sysconf() with the _SC_OPEN_MAX option.

Errors
No errors are defined.
getgrent_r

Name

getgrent_r — reentrantly get entry in group file

Synopsis

#include <grp.h>
int getgrent_r(struct group * gbuf, char * buf, size_t buflen, struct group ** gbufp);

Description

The reentrant interface getgrent_r() shall function in the same way as the interface
getgrent(), except that getgrent_r() shall return the group name, group password,
and group members in buffers provided by the caller, rather than as a pointer to static
storage.

The parameter gbuf contains the struct group that was read from the stream, if any.
The parameter buf contains additional strings, if any.
The parameter buflen specifies the size of buf.
The parameter *gbufp returns a pointer to the struct group in *gbuf.

Return Value

On success, getgrent_r() shall return 0, and *gbufp shall contain a pointer to the
result.

On failure, *gbufp shall contain NULL, and getgrent_r() shall return an error as fol-

Errors

ENOENT

No more group entries.

ERANGE

Not enough buffer space. Specify a larger buffer and try again.
getgrouplist

Name
getgrouplist — get groups a user belongs to

Synopsis
#include <grp.h>
int getgrouplist(const char * user, gid_t group, gid_t * groups, int * ngroups);

Description
The getgrouplist() function shall fill in the array groups with the supplementary groups for the user specified by user. On entry, ngroups shall refer to an integer containing the maximum number of elements in the groups array. The group group shall also be included in the values returned in groups. It is expected that group would be specified as the user's primary group from the password file (obtainable via getpwnam() or a similar function).

Return Value
If on entry the value referenced by ngroups was greater than or equal to the number of supplementary group identifiers to be copied to the array identified by groups, getgrouplist() shall return the number of group identifiers actually copied, and shall set the value referenced by ngroups to this value.

If on entry the value referenced by ngroups was less than the number of supplementary group identifiers, getgrouplist() shall return -1. The initial ngroups entries in groups shall be overwritten.

If the number of groups exceeds the input ngroups value, then as well as returning -1, ngroups shall be set to the number of groups that would have been placed in groups if it had been large enough.

Note: In such a case, the caller can use the information returned to make a further getgrouplist() call with a correctly sized groups array.

If user does not refer to a valid user on the system, then the behavior of this function is undefined.

Errors
None defined.

See Also
groups()
gethostbyaddr_r

Name
gethostbyaddr_r — find network host database entry matching host name (DEPRECATED)

Synopsis
#include <netdb.h>
int gethostbyaddr_r(const void * restrict addr, socklen_t len, int type, struct hostent * restrict result_buf, char * restrict buf, size_t buflen, struct hostent * * restrict result, int * h_errnop);

Description
Note: The gethostbyaddr_r() function is deprecated; applications should use getaddrinfo() instead.

gethostbyaddr_r() is a reentrant version of gethostbyaddr() that searches the network host database for a host address match.

The gethostbyaddr_r() function shall search the network host database for an entry of address family type with the host with address addr. The len argument contains the length of the address referenced by addr.

If type is AF_INET, the addr argument shall be an in_addr structure. If type is AF_INET6, the addr argument shall be an in6_addr structure. If type is any other value, the behavior is unspecified.

The application must provide a buffer for the gethostbyaddr_r() to use during the lookup process. The buffer is referenced by buf, and is of size buflen. If the buffer is not of sufficient size, gethostbyaddr_r() may fail and return ERANGE. If a matching entry is found in the database, gethostbyaddr_r() shall copy the relevant information to the application supplied hostent structure referenced by result_buf, and return a pointer to this structure in *result. If no matching entry is found, *result shall be set to a null pointer. Additional error information shall be set in the variable referenced by h_errnop.

Return Value
On success, the gethostbyaddr_r() function shall return zero. If the return value was ERANGE, the size of the buffer buf, indicated by buflen, was too small. If the gethostbyaddr_r() function returns returns any other value, then the variable referenced by h_errnop shall be set to indicate the cause as for gethostbyaddr().
gethostbyname2

Name

gethostbyname2 — find network host database entry matching host name
(DEPRECATED)

Synopsis

int gethostbyname2(const char * restrict name, int af);

Description

Note: The gethostbyname2() function is deprecated; applications should use
getaddrinfo() instead.

The gethostbyname2() function shall search the network host database for an entry
with name name. This function is similar to the gethostbyname() function but addi-
tionally allows the search to be restricted to a particular address family specified by af.

Return Value

On success, the gethostbyname2() function shall return a pointer to a hostent struc-
ture if the requested entry was found, and a null pointer otherwise.

On unsuccessful completion, gethostbyname2() shall set h_errno as for gethost-

Errors

The gethostbyname2() shall set h_errno as for gethostbyname() in POSIX
gethostbyname2_r

Name

gethostbyname2_r — find network host database entry matching host name
(DEPRECATED)

Synopsis

int gethostbyname2_r(const char * restrict name, int af, struct hostent * restrict result_buf, char * restrict buf, size_t buflen, struct hostent ** restrict result, int * restrict h_errnop);

Description

Note: The gethostbyname2_r() function is deprecated; applications should use getaddrinfo() instead.

The gethostbyname2_r() function shall search the network host database for an entry with name name. gethostbyname2_r() is a reentrant version of gethostbyname2(). These functions are similar to the gethostbyname() and gethostbyname_r() functions but additionally allow the search to be restricted to a particular address family specified by af.

The application must provide a buffer for the gethostbyname2_r() function to use during the lookup process. The buffer is referenced by buf, and is of size buflen. If the buffer is not of sufficient size, gethostbyname_r() may fail and return ERANGE. If a matching entry is found in the database, gethostbyname2_r() shall copy the relevant information to the application-supplied hostent structure referenced by result_buf, and return a pointer to this structure in *result. If no matching entry is found, *result shall be set to a null pointer. Additional error information shall be set in the variable referenced by h_errnop.

Return Value

On success, the gethostbyname2_r() function shall return zero. If the return value was ERANGE, the size of the buffer buf, indicated by buflen, was too small. If the gethostbyname2_r() function returns returns any other value, then the variable referenced by h_errnop shall be set to indicate the cause as for gethostbyname_r().
gethostbyname_r

Name
gethostbyname_r — find network host database entry matching host name
(DEPRECATED)

Synopsis

int gethostbyname_r(const char * restrict name, struct hostent * restrict result_buf, char * restrict buf, size_t buflen, struct hostent ** restrict result, int * restrict h_errno);

Description

Note: The gethostbyname_r() function is deprecated; applications should use getaddrinfo() instead.

gethostbyname_r() is a reentrant version of gethostbyname() that searches the network host database for a host name match.

The gethostbyname_r() function shall search the network host database for an entry with name name.

The application must provide a buffer for the gethostbyname_r() to use during the lookup process. The buffer is referenced by buf, and is of size buflen. If the buffer is not of sufficient size, gethostbyname_r() may fail and return ERANGE. If a matching entry is found in the database, gethostbyname_r() shall copy the relevant information to the application supplied hostent structure referenced by result_buf, and return a pointer to this structure in *result. If no matching entry is found, *result shall be set to a null pointer. Additional error information shall be set in the variable referenced by h_errno.

Return Value

On success, the gethostbyname_r() function shall return zero. If the return value was ERANGE, the size of the buffer buf, indicated by buflen, was too small. If the gethostbyname_r() function returns returns any other value, then the variable referenced by h_errno shall be set to indicate the cause as for gethostbyname().
getifaddrs

Name
getifaddrs, freeifaddrs — get interface addresses

Synopsis
#include <ifaddrs.h>
int getifaddrs(struct ifaddrs ** ifap);
void freeifaddrs(struct ifaddrs * ifa);

Description
The getifaddrs() function creates a linked list of structures describing the network interfaces of the local system. The address of the first item is stored in memory pointed to by ifap. The data returned is dynamically allocated, and should be freed using freeifaddrs().

The list consists of structures of type ifaddrs (see Data Definitions above).

Return Value
On success, getifaddrs() returns zero; on error, -1 is returned, and errno is set appropriately.

Errors
getifaddrs() may fail and set errno for any of the errors specified for socket(), bind(), getsockname(), recvmsg(), sendto(), malloc(), or realloc().

See Also
bind(), getsockname(), socket.

getloadavg

Name
getloadavg — get system load averages

Synopsis
#include <stdlib.h>
int getloadavg(double loadavg[], int nelem);

Description
getloadavg() returns the number of processes in the system run queue averaged over various periods of time. Up to nelem samples are retrieved and assigned to successive elements of loadavg[]. The system imposes a maximum of 3 samples, representing averages over the last 1, 5, and 15 minutes, respectively.

Return Value
If the load average could not be obtained, -1 is returned. Otherwise, the number of samples actually retrieved is returned.
getopt

Name
getopt — parse command line options

Synopsis
#include <unistd.h>
int getopt(int argc, char * const argv[], const char * optstring);

extern char *optarg;
extern int optind, opterr, optopt;

Description

The getopt() function shall parse command line arguments as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), with the following exceptions, where LSB and POSIX specifications vary. LSB systems shall implement the modified behaviors described below.

Argument Ordering

The getopt() function can process command line arguments referenced by argv in one of three ways:

PERMUTE

the order of arguments in argv is altered so that all options (and their arguments) are moved in front of all of the operands. This is the default behavior.

Note: This behavior has undefined results if argv is not modifiable. This is to support historic behavior predating the use of const and ISO C (1999). The function prototype was aligned with POSIX 1003.1-2008 (ISO/IEC 9945-2009) despite the fact that it modifies argv, and the library maintainers are unwilling to change this.

REQUIRE_ORDER

The arguments in argv are processed in exactly the order given, and option processing stops when the first non-option argument is reached, or when the element of argv is "--". This ordering can be enforced either by setting the environment variable POSIXLY_CORRECT, or by setting the first character of optstring to '+'.

RETURN_IN_ORDER

The order of arguments is not altered, and all arguments are processed. Non-option arguments (operands) are handled as if they were the argument to an option with the value 1 ('001'). This ordering is selected by setting the first character of optstring to '-'

Option Characteristics

LSB specifies that:

• an element of argv that starts with "-" (and is not exactly ".-" or "--") is an option element.
• characters of an option element, aside from the initial ",-", are option characters.

POSIX specifies that:

• applications using getopt() shall obey the following syntax guidelines:
  • option name is a single alphanumeric character from the portable character set
  • option is preceded by the ' - ' delimiter character
  • options without option-arguments should be accepted when grouped behind one ' - ' delimiter
  • each option and option-argument is a separate argument
  • option-arguments are not optional
  • all options should precede operands on the command line
  • the argument "--" is accepted as a delimiter indicating the end of options and the consideration of subsequent arguments, if any, as operands
• historical implementations of getopt() support other characters as options as an allowed extension, but applications that use extensions are not maximally portable.
• support for multi-byte option characters is only possible when such characters can be represented as type `int`.

• applications that call any utility with a first operand starting with `-' should usually specify `"-" to mark the end of the options. Standard utilities that do not support this guideline indicate that fact in the OPTIONS section of the utility description.

Extensions

**LSB** specifies that:

• if a character is followed by two colons, the option takes an optional argument; if there is text in the current `argv` element, it is returned in `optarg`, otherwise `optarg` is set to `0`.

• if `optstring` contains `W` followed by a semi-colon (`;`), then `-W foo` is treated as the long option `--foo`.

  **Note:** See `getopt_long()` for a description of long options.

• The first character of `optstring` shall modify the behavior of `getopt()` as follows:
  • if the first character is `+`, then `REQUIRE_ORDER` processing shall be in effect (see above)
  • if the first character is `-`, then `RETURN_IN_ORDER` processing shall be in effect (see above)
  • if the first character is `:`, then `getopt()` shall return `:` instead of `?` to indicate a missing option argument, and shall not print any diagnostic message to `stderr`.

**POSIX** specifies that:

• the `-W` option is reserved for implementation extensions.

Return Values

**LSB** specifies the following additional `getopt()` return values:

• `\001` is returned if `RETURN_IN_ORDER` argument ordering is in effect, and the next argument is an operand, not an option. The argument is available in `optarg`.

Any other return value has the same meaning as for **POSIX**.

**POSIX** specifies the following `getopt()` return values:

• the next option character is returned, if found successfully.

• `:` is returned if a parameter is missing for one of the options and the first character of `optstring` is `:`.

• `?` is returned if an unknown option character not in `optstring` is encountered, or if `getopt()` detects a missing argument and the first character of `optstring` is not `:`.

• `-1` is returned for the end of the option list.

Environment Variables

**LSB** specifies that:

• if the variable `POSIXLY_CORRECT` is set, option processing stops as soon as a non-option argument is encountered.

• the variable `_[PID]_GNU_nonoption_argv_flags_` (where `[PID]` is the process ID for the current process), contains a space separated list of arguments that should not be treated as arguments even though they appear to be so.

  **Rationale:** This was used by bash 2.0 to communicate to GNU libc which arguments resulted from wildcard expansion and so should not be considered as options. This behavior was removed in bash version 2.01, but the support remains in GNU libc.
This behavior is DEPRECATED in this version of the LSB; future revisions of this specification may not include this requirement.

getopt_long

Name
getopt_long — parse command line options

Synopsis

#define _GNU_SOURCE
#include <getopt.h>
int getopt_long(int argc, char * const argv[], const char * opstring,
const struct option * longopts, int * longindex);

Description

gopt long() works like getopt() except that it also accepts long options, started out by two dashes. Long option names may be abbreviated if the abbreviation is unique or is an exact match for some defined option. A long option may take a parameter, of the form --arg=param or --arg param.

longopts is a pointer to the first element of an array of struct opton declared in getopt.h as:

    struct option {
        const char *name;
        int has_arg;
        int *flag;
        int val;
    };

The fields in this structure have the following meaning:

name

The name of the long option.

has_arg

One of:
no_argument (or 0) if the option does not take an argument,
required_argument (or 1) if the option requires an argument, or
optional_argument (or 2) if the option takes an optional argument.

flag

specifies how results are returned for a long option. If flag is NULL, then
getopt_long() shall return val. (For example, the calling program may set val
to the equivalent short option character.) Otherwise, getopt_long() returns 0,
and flag shall point to a variable which shall be set to val if the option is found,
but left unchanged if the option is not found.

val

The value to return, or to load into the variable pointed to by flag.
If longindex is not NULL, it points to a variable which is set to the index of the long
option relative to longopts.

Return Value

getopt_long() returns the option character if a short option was found successfully,
or ":" if there was a missing parameter for one of the options, or "?" for an unknown op-
tion character, or -1 for the end of the option list.

For a long option, getopt_long() returns val if flag is NULL, and 0 otherwise. Error
and -1 returns are the same as for getopt(), plus "?" for an ambiguous match or an ex-
traneous parameter.

getopt_long_only

Name

g getopt_long_only — parse command line options

Synopsis

#define GNU_SOURCE
#include <unistd.h>
int getpagesize(void);

Description
getpagesize() returns the number of bytes in a memory page.

Note: The getpagesize() function is deprecated. Portable applications should use sysconf(_SC_PAGE_SIZE) instead.

Return Value
The getpagesize() function returns the current page size.

Errors
No errors are defined.
getprotobyname_r

Name
getprotobyname_r — retrieve information from the network protocol database by protocol name, reentranly

Synopsis
#include <netdb.h>
int getprotobyname_r(const char * name, struct protoent * result_buf, char * buf, size_t buflen, struct protoent ** result);

Description
The getprotobyname_r() function is a reentrant version of the getprotobyname() function.

The getprotobyname_r() function shall search the network protocol database for an entry with the name name.

If a matching entry is found in the database, this function shall copy the relevant information to the application-supplied protoent structure referenced by result_buf, and return a pointer to this structure in *result. If no matching entry is found, *result shall be set to a null pointer.

The array buf shall contain the string fields referenced by the protoent structure that was returned. The parameter buflen shall specify the array's size. 1024 bytes should be enough for most uses.

Return Value
On success, the getprotobyname_r() function shall return 0. If the return value was ERANGE, the size of the buffer buf, indicated by buflen, was too small.
getprotobynumber_r

Name
getprotobynumber_r — retrieve information from the network protocol database by protocol number, reentrantly

Synopsis
#include <netdb.h>
int getprotobynumber_r(int proto, struct protoent * result_buf, char * buf, size_t buflen, struct protoent * * result);

Description
The getprotobynumber_r() function is a reentrant version of the getprotobynumber() function.

The getprotobynumber_r() function shall search the network protocol database for an entry with protocol number proto.

If a matching entry is found in the database, this function shall copy the relevant information to the application-supplied protoent structure referenced by result_buf, and return a pointer to this structure in *result. If no matching entry is found, *result shall be set to a null pointer.

The array buf shall contain the string fields referenced by the protoent structure that was returned. The parameter buflen shall specify the array's size. 1024 bytes should be enough for most uses.

Return Value
On success, the getprotobynumber_r() function shall return 0. If the return value was ERANGE, the size of the buffer buf, indicated by buflen, was too small.
getprotoent_r

Name
getprotoent_r — read the next entry of the protocol database, reentrantly

Synopsis
#include <netdb.h>
int getprotoent_r(struct protoent * result_buf, char * buf, size_t buflen, struct protoent ** result);

Description
The getprotoent_r() function is a reentrant version of the getprotoent() function.
The getprotoent_r() function shall search the network protocol database for the next entry.
If the next entry is found in the database, this function shall copy the relevant information to the application-supplied protoent structure referenced by result_buf, and return a pointer to this structure in *result. If no next entry is found, *result shall be set to a null pointer.
The array buf shall contain the string fields referenced by the protoent structure that was returned. The parameter buflen shall specify the array’s size. 1024 bytes should be enough for most uses.

Return Value
On success, the getprotoent_r() function shall return zero.
If the return value was ENOENT, there were no more entries in the database.
If the return value was ERANGE, the size of the buffer buf, indicated by buflen, was too small.
getpwent_r

Name
getpwent_r — reentrantly get entry in passwd file

Synopsis
#include <pwd.h>
int getpwent_r(struct passwd * pwbuf, char * buf, size_t buflen,
struct passwd ** pwbufp);

Description
The reentrant interface getpwent_r() shall function in the same way as the interface
getpwent(), except that getpwent_r() shall return the user name, user password,
GECOS field, home directory, and shell program in buffers provided by the caller,
rather than as a pointer to static storage.
The parameter pwbuf contains the struct passwd that was read from the stream, if any.
The parameter buf contains additional strings, if any.
The parameter buflen specifies the size of buf.
The parameter *pwbufp returns a pointer to the struct passwd in *pwbuf.

Return Value
On success, getpwent_r() shall return 0, and *pwbufp shall contain a pointer to the result.
On failure, *pwbufp shall contain NULL, and getpwent_r() shall return an error as follows.

Errors
ENOENT
No more password entries.
ERANGE
Not enough buffer space. Specify a larger buffer and try again.
getrlimit, setrlimit

Name
getrlimit, setrlimit — get resource consumption limits

Synopsis
#include <sys/resource.h>
int getrlimit(__rlimit_resource_t __resource, struct rlimit * __rlimits);
int setrlimit(__rlimit_resource_t __resource, const struct rlimit * __rlimits);

Description
getrlimit() and setrlimit() are as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

Extra Resources
These additional resources extend the list in POSIX 1003.1-2008 (ISO/IEC 9945-2009).

RLIMIT_NPROC
The maximum number of processes (or, more precisely on Linux, threads) that can be created for the real user ID of the calling process. Upon encountering this limit, fork() shall fail with the error EAGAIN.

RLIMIT_MEMLOCK
The maximum number of bytes of memory that may be locked into RAM. In effect this limit is rounded down to the nearest multiple of the system page size. This limit affects mlock() and mlockall(), the mmap() MAP_LOCKED operation and the shmctl() SHM_LOCK operation. The shmctl() SHM_LOCK locks are accounted for separately from the per-process memory locks established by mlock(), mlockall(), and mmap() MAP_LOCKED. In the former case, the limit sets a maximum on the total bytes in shared memory segments (see shmget()) that may be locked by the real user ID of the calling process. A process can lock bytes up to this limit in each of these two categories.

RLIMIT_LOCKS
A limit on the combined number of flock() locks and fcntl() leases that this process may establish. This limit is obsolete and should not be used; support depends heavily on the version of the operating system kernel.

RLIMIT_RSS
Specifies the limit (in pages) of the process's resident set. This limit is obsolete and should not be used; support depends heavily on the version of the operating system kernel. It affects only calls to madvise() specifying MADV_WILLNEED.

RLIMIT_SIGPENDING
Specifies the limit on the number of signals that may be queued for the real user ID of the calling process. Both standard and real-time signals are counted for the purpose of checking this limit. However, the limit is enforced only for sigqueue(); it is always possible to use kill() to queue one instance of any of the signals that are not already queued to the process.

RLIMIT_MSGQUEUE
LSB Core - Generic 5.0

Specifies the limit on the number of bytes that can be allocated for POSIX message queues for the real user ID of the calling process. This limit is enforced for `mq_open()`. Each message queue that the user creates counts (until it is removed) against this limit according to the formula:

\[
\text{bytes} = \text{attr.mq_maxmsg} \times \text{sizeof(struct msg_msg *)} + \text{attr.mq_maxmsg} \times \text{attr.mq_msgsize}
\]

where `attr` is the `mq_attr` structure specified as the fourth argument to `mq_open(3)`.

The first addend in the formula, which includes `sizeof(struct msg_msg *)` (4 bytes on Linux/i386), ensures that the user cannot create an unlimited number of zero-length messages (such messages nevertheless each consume some system memory for bookkeeping overhead).

**RLIMIT_NICE**

Specifies a ceiling to which the process's nice value can be raised using `setpriority()` or `nice()`. The actual ceiling for the nice value is calculated as 20 minus the value of `rlim_cur`.

**RLIMIT_RTPRIO**

Specifies a ceiling on the real-time priority that may be set for this process using `sched_setscheduler(2)` and `sched_setparam(2)`.

**RLIMIT_RTTIME**

Specifies a limit (in microseconds) on the amount of CPU time that a process scheduled under a real-time scheduling policy may consume without making a blocking system call. For the purpose of this limit, each time a process makes a blocking system call, the count of its consumed CPU time is reset to zero. The CPU time count is not reset if the process continues trying to use the CPU but is preempted, its time slice expires, or it calls `sched_yield()`.

Upon reaching the soft limit, the process is sent a `SIGXCPU` signal. If the process catches or ignores this signal and continues consuming CPU time, then `SIGXCPU` will be generated once each second until the hard limit is reached, at which point the process is sent a `SIGKILL` signal.

The intended use of this limit is to stop a runaway real-time process from locking up the system.

**Extra Errors**

These additional error codes extend the list in `POSIX.1003.1-2008 (ISO/IEC 9945-2009)`.

**EFAULT**

A pointer argument points to a location outside the accessible address space.
getservbyname_r

Name

The getservbyname_r() function is a reentrant version of the getservbyname() function.

The getservbyname_r() function shall search the network services database for an entry with the name name. The proto parameter shall restrict the search to entries with the specified protocol. If proto is NULL, getservbyname_r() may return entries with any protocol.

If a matching entry is found in the database, this function shall copy the relevant information to the application-supplied servent structure referenced by result_buf, and return a pointer to this structure in *result. If no matching entry is found, *result shall be set to a null pointer.

The array buf shall contain the string fields referenced by the servent structure that was returned. The parameter buflen shall specify the array's size. 1024 bytes should be enough for most uses.

Return Value

On success, the getservbyname_r() function shall return zero. If the return value was ERANGE, the size of the buffer buf, indicated by buflen, was too small.
getservbyport_r

Name
getservbyport_r — retrieve information from the network services database by service port, reentrantly

Synopsis
#include <netdb.h>
int getservbyport_r(int port, const char *proto, struct servent *result_buf, char *buf, size_t buflen, struct servent **result);

Description
The getservbyport_r() function is a reentrant version of the getservbyport() function.

The getservbyport_r() function shall search the network services database for an entry with the port port. The proto parameter shall restrict the search to entries with the specified protocol. If proto is NULL, getservbyport_r() may return entries with any protocol.

If a matching entry is found in the database, this function shall copy the relevant information to the application-supplied servent structure referenced by result_buf, and return a pointer to this structure in *result. If no matching entry is found, *result shall be set to a null pointer.

The array buf shall contain the string fields referenced by the servent structure that was returned. The parameter buflen shall specify the array's size. 1024 bytes should be enough for most uses.

Return Value
On success, the getservbyport_r() function shall return zero. If the return value was ERANGE, the size of the buffer buf, indicated by buflen, was too small.
getservent_r

Name
getservent_r — read the next entry of the network services database, reentrantly

Synopsis
#include <netdb.h>
int getservent_r(struct servent * result_buf, char * buf, size_t buflen, struct servent ** result);

Description
The getservent_r() function is a reentrant version of the getservent() function.
The getservent_r() function shall search the network services database for the next entry.
If the next entry is found in the database, this function shall copy the relevant information to the application-supplied servent structure referenced by result_buf, and return a pointer to this structure in *result. If no next entry is found, *result shall be set to a null pointer.
The array buf shall contain the string fields referenced by the servent structure that was returned. The parameter buflen shall specify the array’s size. 1024 bytes should be enough for most uses.

Return Value
On success, the getservent_r() function shall return 0.
If the return value was ENOENT, there were no more entries in the database.
If the return value was ERANGE, the size of the buffer buf, indicated by buflen, was too small.

getsockopt

Name
getsockopt — get socket options

Synopsis
#include <sys/socket.h>
#include <netinet/ip.h>
int getsockopt(int socket, int level, int option_name, void * restrict option_value, socklen_t * restrict option_len);

**Description**

The `getsockopt()` function shall behave as specified in `POSIX.1003.1-2008` (ISO/IEC 9945-2009), with the following extensions.

**IP Protocol Level Options**

If the `level` parameter is `IPPROTO_IP`, the following values shall be supported for `option_name` (see `RFC 791:Internet Protocol` for further details):

**IP_OPTIONS**

Get the Internet Protocol options sent with every packet from this socket. The `option_value` shall point to a memory buffer in which the options shall be placed; on entry `option_len` shall point to an integer value indicating the maximum size of the memory buffer, in bytes. On successful return, the value referenced by `option_len` shall be updated to the size of data copied to the buffer. For IPv4, the maximum length of options is 40 bytes.

**IP_TTL**

Get the current unicast Internet Protocol Time To Live value used when sending packets with this socket. The `option_value` shall point to a buffer large enough to hold the time to live value (at least 1 byte), and `option_len` shall point to an integer value holding the maximum size of that buffer. On successful return, the value referenced by `option_len` shall be updated to contain the number of bytes copied into the buffer, which shall be no larger than the initial value, and `option_value` shall point to an integer containing the time to live value.

**IP_TOS**

Get the Internet Protocol type of service indicator used when sending packets with this socket. The `option_value` shall point to a buffer large enough to hold the type of service indicator (at least 1 byte), and `option_len` shall point to an integer value holding the maximum size of that buffer. On successful return, the value referenced by `option_len` shall be updated to contain the number of bytes copied into the buffer, which shall be no larger than the initial value, and `option_value` shall point to an integer containing the time to live value.
gettext

Name
ggettext — search message catalogs for a string

Synopsis

#include <libintl.h>
char * gettext(const char * msgid);

Description

The gettext() function shall search the currently selected message catalogs for a string identified by the string msgid. If a string is located, that string shall be returned.

The gettext() function is equivalent to dcgettext(NULL, msgid, LC_MESSAGES).

Return Value

If a string is found in the currently selected message catalogs for msgid, then a pointer to that string shall be returned. Otherwise, a pointer to msgid shall be returned.

Applications shall not modify the string returned by gettext().

Errors

None.

The gettext() function shall not modify errno.

See Also

dgettext, ngettext, dgettext, dcgettext, dcngettext, textdomain, bindtextdomain, bind_textdomain_codeset
getutent

Name
getutent — access user accounting database entries

Synopsis
#include <utmp.h>
struct utmp *getutent(void);

Description
The getutent() function shall read the next entry from the user accounting database.

Return Value
Upon successful completion, getutent() shall return a pointer to a utmp structure containing a copy of the requested entry in the user accounting database. Otherwise, a null pointer shall be returned. The return value may point to a static area which is over-written by a subsequent call to getutent().

Errors
None defined.

getutent_r

Name
getutent_r — access user accounting database entries

Synopsis
int getutent_r(struct utmp * buffer, struct utmp ** result);

Description
The getutent_r() function is a reentrant version of the getutent() function. On entry, buffer should point to a user supplied buffer to which the next entry in the database will be copied, and result should point to a location where the result will be stored.

Return Value
On success, getutent_r() shall return 0 and set the location referenced by result to a pointer to buffer. Otherwise, getutent_r() shall return -1 and set the location referenced by result to NULL.

getwc_unlocked

Name
getc_unlocked — non-thread-safe getwc

Description
getc_unlocked() is the same as getwc(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().
**getwchar_unlocked**

**Name**

getwchar_unlocked — non-thread-safe getwchar

**Description**

getwchar_unlocked() is the same as getwchar(), except that it need not be thread-safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().

**glob64**

**Name**

glob64 — find pathnames matching a pattern (Large File Support)

**Synopsis**

```
#include <glob.h>
int glob64(const char * pattern, int flags, int (*errfunc) (const char *, int), glob64_t * pglob);
```

**Description**

glob64() is a large-file version of the glob() function defined in POSIX 1003.1-2008 (ISO/IEC 9945-2009). It shall search for pathnames matching pattern according to the rules used by the shell, /bin/sh. No tilde expansion or parameter substitution is done; see wordexp().

The results of a glob64() call are stored in the structure pointed to by pglob, which is a glob64_t declared in glob.h with the following members:

```c
typedef struct
{
    size_t gl_pathc;
    char **gl_pathv;
    size_t gl_offs;
    int gl_flags;
    void (*gl_closedir) (void *);
    struct dirent64 *(*gl_readdir64) (void *);
    void *(*gl_opendir) (const char *);
    int (*gl_lstat) (const char *, struct stat *);
    int (*gl_stat) (const char *, struct stat *);
} 306
```
glob64_t;

Structure members with the same name as corresponding members of a glob_t as defined in POSIX 1003.1-2008 (ISO/IEC 9945-2009) shall have the same purpose.

Other members are defined as follows:

gl_flags
   reserved for internal use

gl_closedir
   pointer to a function capable of closing a directory opened by gl_opendir

gl_readdir64
   pointer to a function capable of reading entries in a large directory

gl_opendir
   pointer to a function capable of opening a large directory

gl_stat
   pointer to a function capable of returning file status for a large file

gl_lstat
   pointer to a function capable of returning file status information for a large file or symbolic link

A large file or large directory is one with a size which cannot be represented by a variable of type off_t.

**Return Value**

On success, 0 is returned. Other possible returns are:

GLOB_NOSPACE
   out of memory

GLOB_ABORTED
   read error

GLOB_NOMATCH
   no match found
globfree64

Name

globfree64 — free memory from glob64() (Large File Support)

Synopsis

#include <glob.h>
void globfree64(glob64_t * pglob);

Description

globfree64() frees the dynamically allocated storage from an earlier call to

glob64().

globfree64() is a large-file version of the globfree() function defined in POSIX


gnu_get_libc_version, gnu_get_libc_release

Name

gnu_get_libc_version, gnu_get_libc_release — get glibc-specific

version and release

Synopsis

#include <gnu/libc-version.h>
const char * gnu_get_libc_version(void);
const char * gnu_get_libc_release(void);

Description

gnu_get_libc_version() returns a string that identifies the version of the C library

running the program making the call.

gnu_get_libc_release() returns a string indicates the release status of the C library

running the program making the call. This will be a string such as "stable".

Return Value

The functions return strings. The contents of these strings are unspecified.

Errors

No errors are defined.

Notes

These functions are specific to GNU libc (glibc). This specification does not require the
implementation of libc to be glibc, although it requires these functions.

The string returned by gnu_get_libc_version() will be a dotted version string, which may have meaning to developers otherwise familiar with glibc. These functions
have been requested to aid in portability of software which also runs in non-LSB con-
texts, but decisions based on the return value should be tempered by an understanding of
what the behavioral requirements of this specification are. That is, it may or may not be
useful to discover that a running system, for example, has version "2.10.1" if that im-
plies different behavior than described by this specification.
hcreate_r

Name
hcreate_r — allocate space for a hash search table, reentrantly

Synopsis
#include <search.h>
int hcreate_r(size_t nel, struct hsearch_data * htab);

Description
The hcreate_r() function is a reentrant version of the hcreate() function.

hcreate_r() shall initialize the object referenced by htab with a hash table containing at least nel elements. Unlike its non-reentrant equivalent, hcreate(), the hcreate_r() function may work with more than one hash table.

The memory for the htab object may be dynamically allocated. It must be initialized with 0 before hcreate_r() is called.

Return Value
On success, hcreate_r() shall return a non-zero value.
On failure, hcreate_r() shall return 0. This usually happens because not enough memory was available.

hdestroy_r

Name
hdestroy_r — dispose of a hash search table, reentrantly

Synopsis
#include <search.h>
void hdestroy_r(struct hsearch_data * htab);

Description
The hdestroy_r() function is a reentrant version of the hdestroy() function.

hdestroy_r() frees the resources allocated by hcreate_r() for the object htab.
hsearch_r

Name
hsearch_r — search a hash table, reentrantly

Synopsis
#include <search.h>
int hsearch_r(ENTRY item, ACTION action, ENTRY * * retval, struct hsearch_data * htab);

Description
The hsearch_r() is a reentrant version of the hsearch() function, but instead of operating on a single global hash table, hsearch_r() operates on the table described by the object that htab references. This object can be initialized with the function hcreate_r().

Unlike the hsearch() function, hsearch_r() returns a pointer to the found entry in the variable referred to by retval, rather than directly.

Return Value
On success, hsearch_r() shall return a non-zero value.
On failure, hsearch_r() shall return 0 and set errno to an appropriate value.

Errors
ENOMEM
action was set to ENTER, but the table was full.

ESRCH
action was set to FIND, but no matching element was found in the table.

inet_aton

Name
inet_aton — Internet address manipulation routine

Synopsis
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
int inet_aton(const char * cp, struct in_addr * inp);

Description
inet_aton() converts the Internet host address cp from the standard IPv4 numbers-and-dots notation into binary data and stores it in the structure that inp points to.
inet_aton() returns a nonzero value if the address is valid, 0 if not.

Note: Note that on some LSB architectures, the host byte order is Least Significant Byte first, whereas the network byte order, as used on the Internet, is Most Significant Byte first.
initgroups

Name
initgroups — initialize the supplementary group access list

Synopsis

```
#include <grp.h>
#include <sys/types.h>
int initgroups(const char * user, gid_t group);
```

Description

If the process has appropriate privilege, the initgroups() function shall initialize the Supplementary Group IDs for the current process by reading the group database and using all groups of which user is a member. The additional group group is also added to the list.

Return Value

On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

Errors

EPERM
The calling process does not have sufficient privileges.

ENOMEM
Insufficient memory to allocate group information structure.

See Also

setgroups()

initstate_r

Name
initstate_r — reentrantly initialize a state array for random number generator functions

Synopsis

```
#include <stdlib.h>
int initstate_r(unsigned int seed, char * statebuf, size_t statelen, struct random_data * buffer);
```

Description

The interface initstate_r() shall function in the same way as the interface initstate(), except that initstate_r() shall use the data in buffer instead of the global random number generator state.
**inotify_add_watch**

**Name**

`inotify_add_watch` — add a watch to a watch list

**Synopsis**

```c
#include <sys/inotify.h>
int inotify_add_watch(int fd, const char *path, uint32_t mask);
```

**Description**

`inotify_add_watch()` shall add a watch to, or modify an existing watch on, the watch list of the inotify instance specified by the file descriptor `fd`, for the file specified by `path`, to monitor the events specified by the bitmask `mask`. The caller must have read access to the file.

**Return Value**

On success, `inotify_add_watch()` shall return the unique, non-negative watch descriptor associated with the file `path` and the inotify instance specified by the file descriptor `fd`.

If `path` was already on the watch list, then `inotify_add_watch()` shall return the existing watch descriptor.

If `path` was not already on the watch list, then `inotify_add_watch()` shall allocate a new watch descriptor.

`inotify_add_watch()` shall not work recursively. Monitoring subdirectories of `path` shall require adding watches to them.

On failure, `inotify_add_watch()` shall return -1 and set `errno` to an appropriate value.

**Errors**

EACCESS

The caller does not have read access to `path`.

EBADF

The file descriptor `fd` is invalid.

EFAULT

`path` is outside of the address space accessible by the process.

EINVAL

`mask` contains no legal events, or `fd` is not a valid inotify file descriptor.

ENOMEM

There is not enough kernel memory available.

ENOSPC

The maximum number of watches has been created for this user, or the kernel cannot allocate a resource.

**Application Usage**
Reading

The function `read()` can be used to determine which inotify events have occurred. A blocking file descriptor will make `read()` block until at least one event has occurred.

If successful, `read()` will return at least one of the following `inotify_event` structures in a buffer:

```c
struct inotify_event {
    int wd;
    uint32_t mask;
    uint32_t cookie;
    uint32_t len;
    char path[];
};
```

`wd` is a watch descriptor that specifies the watch associated with the event. It is obtained from a previous invocation of `inotify_add_watch()`.

`mask` is a bit mask describing inotify events. See the section on masks below.

`cookie` is an integer associating related inotify events. The integer value is unique, and currently only enables the application to associate `IN_MOVE_FROM` and `IN_MOVE_TO` rename events.

`len` is a count of the bytes in `path`, including null bytes. This means that the total length of an `inotify_event` structure is

```
sizeof(inotify_event)+len
```
path is only returned when an event occurs for a file within a watched directory. This string is null-terminated, and it may contain more null bytes so that future reads will be aligned properly on an address boundary.

In kernels before 2.6.21, read() returns 0 when the buffer given to it is too small to return data about the next event. In subsequent kernels, it fails with the error EINVAL.

For a given file descriptor, the inotify events are returned in an ordered queue. Events on a file descriptor will always be returned in the correct order of occurrence. If two or more inotify events for a given file descriptor have identical values for all fields, then only one inotify_event will be returned to represent all of them.

The number of bytes that can be read from an inotify file descriptor can be determined by making a FIONREAD ioctl() call.

Masks

The mask argument of inotify_add_watch() and the mask field of the inotify_event structure are bit masks that specify inotify events. The bits in the list below can be set in the mask argument of inotify_add_watch() and returned in the mask field of inotify_event.

IN_ACCESS
  File was read.

IN_ALL_EVENTS
  Bit mask of all events in this list.

IN_ATTRIB
  File's metadata changed (including timestamps and permissions).

IN_CLOSE
  Same as

  IN_CLOSE_WRITE | IN_CLOSE_NOWRITE
IN_CLOSE_WRITE
   File that was opened for writing was closed.

IN_CLOSE_NOWRITE
   File that was not opened for writing was closed.

IN_CREATE
   File or directory was created in a watched directory.

IN_DELETE
   File or directory was deleted in a watched directory.

IN_DELETE_SELF
   Watched file or directory was deleted.

IN_MODIFY
   File was changed.

IN_MOVE
   Same as

   IN_MOVED_FROM    | IN_MOVED_TO
IN_MOVE_SELF
   Watched file or directory was moved

IN_MOVED_FROM
   File was moved out of watched directory.

IN_MOVED_TO
   File was moved into watched directory.

IN_OPEN
   File was opened.

All of the events above, except for IN_DELETE_SELF and IN_MOVE_SELF, cause the name field of the inotify_event structure to contain the name of the file or directory being monitored.

The following bit is valid for inotify_add_watch() only.

IN_ONESHOT
   Monitor path for an event, and then remove it from the watch list.

The following bits are valid for the inotify_event structure only.

IN_IGNORED
   Watch was removed, either explicitly (via inotify_rm_watch()) or implicitly (file deletion or file system unmounting).

IN_ISDIR
   Object being watched is a directory.

IN_Q_OVERFLOW
   The event queue overflowed (wd is set to -1).

IN_UNMOUNT
   File system of object being watched was unmounted.

Notes
It is possible to monitor file descriptors with the functions epoll(), poll(), and select().

When all of the file descriptors that point to an inotify instance have been closed, the instance and its associated resources and watches are freed by the kernel.

See Also
inotify_init(), inotify_rm_watch()
inotify_init

Name
inotify_init — instantiate inotify

Synopsis
#include <sys/inotify.h>
int inotify_init(void);

Description
inotify_init() shall create one instance of inotify.

Return Value
On success, inotify_init() shall return a file descriptor pointing to the new inotify instance.
On failure, inotify_init() shall return -1 and set errno to an appropriate value.

Errors
EMFILE
The maximum number of inotify instances has been created for this user.
ENFILE
The maximum number of file descriptors has been created on the system.
ENOMEM
There is not enough kernel memory available.

See Also
inotify_add_watch(), inotify_rm_watch()
**inotify_rm_watch**

**Name**
inotify_rm_watch — remove a watch from an inotify watch list

**Synopsis**
```
#include <sys/inotify.h>
int inotify_rm_watch(int fd, int wd);
```

**Description**
inotify_rm_watch() shall remove the watch associated with the watch descriptor *wd* from the watch list of the inotify instance associated with the file descriptor *fd*.

If a watch is removed, its watch descriptor shall generate the IN_IGNORED event.

**Return Value**
On success, inotify_rm_watch() shall return 0.
On failure, inotify_rm_watch() shall return -1 and set *errno* to an appropriate value.

**Errors**

EBADF

The file descriptor *fd* is invalid.

EINVAL

*wd* is invalid, or *fd* is not a valid inotify file descriptor.

**See Also**
inotify_add_watch(), inotify_init()
ioctl

Name
ioctl — control device

Synopsis
#include <sys/ioctl.h>
int ioctl (int fildes, int request, ...);

Description
The ioctl() function shall manipulate the underlying device parameters of special files. fildes shall be an open file descriptor referring to a special file. The ioctl() function shall take three parameters; the type and value of the third parameter is dependent on the device and request.

Conforming LSB applications shall not call ioctl() except in situations explicitly stated in this specification.

Return Value
On success, 0 is returned. An ioctl() may use the return value as an output parameter and return a non-negative value on success. On error, -1 is returned and the global variable errno is set appropriately.

Errors
EBADF
   fildes is not a valid descriptor.

EFAULT
   The third parameter references an inaccessible memory area.

ENOTTY
   fildes is not associated with a character special device.

ENOTTY
   The specified request does not apply to the kind of object that fildes references.

EINVAL
   request or the third parameter is not valid.

Relationship to POSIX (Informative)
It should be noted that POSIX 1003.1-2008 (ISO/IEC 9945-2009) contains an interface named ioctl(). The LSB only defines behavior when fildes refers to a socket (see sockio) or terminal device (see ttyio), while POSIX 1003.1-2008 (ISO/IEC 9945-2009) only defines behavior when fildes refers to a STREAMS device. An implementation may support both behaviors; the LSB does not require any STREAMS support.
sockio

Name
sockio — socket ioctl commands

Synopsis
#include <sys/ioctl.h>
#include <sys/socket.h>
#include <net/if.h>
#include <netinet/in.h>

int ioctl(int sockfd, int request, void *argp);

## Description

Socket `ioctl()` commands are a subset of the `ioctl()` calls, which can perform a variety of functions on sockets. `sockfd` shall be an open file descriptor referring to a socket (see the `socket()` or `accept()` functions).

Socket `ioctl()` commands apply to the underlying network interfaces, and affect the entire system, not just the file descriptor used to issue the `ioctl()`.

The following values for `request` are accepted:

### SIOCGIFCONF (Deprecated)

Get the interface configuration list for the system.

**Note:** The `SIOCGIFCONF` interface is superseded by the `if_nameindex()` family of functions (see [POSIX 1003.1-2008 (ISO/IEC 9945-2009)]). A future version of this specification may withdraw this value for `request`.

`argp` shall point to a `ifconf` structure, as described in `<net/if.h>`. Before calling, the caller shall set the `ifc_ifcu.ifcu_req` field to point to an array of `ifreq` structures, and set `ifc_len` to the size in bytes of this allocated array. Upon return, `ifc_len` will contain the size in bytes of the array which was actually used. If it is the same as the length upon calling, the caller should assume that the array was too small and try again with a larger array.

On success, `SIOCGIFCONF` shall return a nonnegative value.

**Rationale:** Historical UNIX systems disagree on the meaning of the return value.

### SIOCGIFFLAGS

Get the interface flags for the indicated interface. `argp` shall point to a `ifreq` structure. Before calling, the caller should fill in the `ifr_name` field with the interface name, and upon return, the `ifr_ifru.ifru_flags` field is set with the interface flags.

### SIOCGIFADDR

Get the interface address for the given interface. `argp` shall point to a `ifreq` structure. Before calling, the caller should fill in the `ifr_name` field with the interface name, and upon return, the `ifr_ifru.ifru_addr` field is set with the interface address.

### SIOCGIFBRDADDR

Get the interface broadcast address for the given interface. `argp` shall point to a `ifreq` structure. Before calling, the caller should fill in the `ifr_name` field with the interface name, and upon return, the `ifr_ifru.ifru_broadcast` field is set with the interface broadcast address.

### SIOCGIFDSTADDR

Get the point-to-point address for the given interface. `argp` shall point to a `ifreq` structure. Before calling, the caller should fill in the `ifr_name` field with the interface name, and upon return, the `ifr_dstaddr` field is set with the point-to-point address.

### SIOCGIFNAME

© 2015 Linux Foundation
Get the name of an interface. `argp` shall point to a `ifreq` structure. Before calling, the caller should fill in the `ifr_ifindex` field with the number (index) of the interface, and upon return, the `ifr_name` field is set with the interface name.

**SIOCGIFNETMASK**

Get the network mask for the given interface. `argp` shall point to a `ifreq` structure. Before calling, the caller should fill in the `ifr_name` field with the interface name, and upon return, the `ifr_ifru.ifru_netmask` field is set with the network mask.

**SIOCGIFMTU**

Get the Maximum Transmission Unit (MTU) size for the given interface. `argp` shall point to a `ifreq` structure. Before calling, the caller should fill in the `ifr_name` field with the interface name, and upon return, the `ifr_ifru.ifru_mtu` field is set with the MTU. Note: The range of valid values for MTU varies for an interface depending on the interface type.

**FIONREAD**

Get the amount of queued unread data in the receive buffer. `argp` shall point to an integer where the result is to be placed.

**Note:** Some implementations may also support the use of `FIONREAD` on other types of file descriptor. However, the LSB only specifies its behavior for a socket related file descriptor.

**Return Value**

On success, if `request` is `SIOCGIFCONF`, a non-negative integer shall be returned. If `request` is not `SIOCGIFCONF`, on success 0 is returned. On error, -1 is returned and the global variable `errno` is set appropriately.

**Errors**

**EBADF**

`sockfd` is not a valid descriptor.

**EFAULT**

`argp` references an inaccessible memory area.

**ENOTTY**

The specified `request` does not apply to the kind of object that the descriptor `sockfd` references.

**EINVAL**

Either `request` or `argp` is invalid.

**ENOTCONN**

The operation is only defined on a connected socket, but the socket wasn’t connected.
**ttyio**

**Name**

ttyio — tty ioctl commands

**Synopsis**

```c
#include <sys/ioctl.h>
#include <fcntl.h>
int ioctl(int fd, unsigned long request, int *argp);
```

**Description**

Tty ioctl commands are a subset of the ioctl() calls, which can perform a variety of functions on tty devices. `fd` shall be an open file descriptor referring to a terminal device.

The following ioctl()s are provided:

**TIOCGWINSZ**

Get the size attributes of the terminal or pseudo-terminal identified by `fd`. On entry, `argp` shall reference a `winsize` structure. On return, the structure will have `ws_row` set to the number of rows of text (i.e. lines of text) that can be viewed on the device, and `ws_col` set to the number of columns (i.e. text width).

**Note:** The number of columns stored in `ws_col` assumes that the terminal device is using a mono-spaced font.

**TIOCSWINSZ**

Sets the size attributes of the terminal or pseudo-terminal identified by `fd`. On entry, `argp` shall reference a `winsize` structure. The value of the `winsize` structure's element `ws_row` shall be the number of rows of text (i.e. lines of text) that can be viewed on the device, and the element `ws_col` shall be the number of columns (i.e. text width). Note that this call merely sets the size attributes for the kernel driver, not the window size itself, and is intended to be used to update the kernel driver when the window size is changed.

**Return Value**

On success, 0 is returned. On error, -1 is returned and the global variable `errno` is set appropriately.

**Errors**

EBADF

`fd` is not a valid descriptor.

EFAULT

`argp` references an inaccessible memory area.

EINVAL

`request` and `argp` are not valid.
jrand48_r

Name
jrand48_r — reentrantly generate pseudorandom numbers in a uniform distribution

Synopsis
#include <stdlib.h>
int jrand48_r(unsigned short[3] xsubi, struct drand48_data * buffer,
long int * result);

Description
The interface jrand48_r() shall function in the same way as the interface jrand48(), except that jrand48_r() shall use the data in buffer instead of the global random number generator state.
Before it is used, buffer must be initialized, for example, by calling lcong48_r(), seed48_r(), or srand48_r(), or by filling it with zeroes.

kill

Name
kill — send a signal

Synopsis
#include <signal.h>
int kill(pid_t pid, int sig);

Description
kill() is as specified in the POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

Process ID -1 doesn’t affect calling process
If pid is specified as -1, sig shall not be sent to the calling process. Other than this, the rules in the POSIX 1003.1-2008 (ISO/IEC 9945-2009) apply.

Rationale: This was a deliberate Linus decision after an unpopular experiment in including the calling process in the 2.5.1 kernel. See "What does it mean to signal everybody?", Linux Weekly News, 20 December 2001, http://lwn.net/2001/1220/kernel.php3
lcong48_r

Name
lcong48_r — reentrantly generate pseudorandom numbers in a uniform distribution

Synopsis
#include <stdlib.h>
int lcong48_r(unsigned short[7] param, struct drand48_data * buffer);

Description
The interface lcong48_r() shall function in the same way as the interface lcong48(), except that lcong48_r() shall use the data in buffer instead of the global random number generator state.

link

Name
link — create a link to a file

Synopsis
#include <unistd.h>
int link(const char * path1, const char * path2);

Description
The link() function shall behave as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except with differences as listed below.

Need Not Follow Symlinks
POSIX 1003.1-2008 (ISO/IEC 9945-2009) specifies that pathname resolution shall follow symbolic links during pathname resolution unless the function is required to act on the symbolic link itself, or certain arguments direct that the function act on the symbolic link itself. The link() function in POSIX 1003.1-2008 (ISO/IEC 9945-2009) contains no such requirement to operate on a symbolic link. However, a conforming LSB implementation need not follow a symbolic link for the path1 argument.
**lrand48_r**

**Name**

lrand48_r — reentrantly generate pseudorandom numbers in a uniform distribution

**Synopsis**

```c
#include <stdlib.h>
int lrand48_r(struct drand48_data * buffer, long int * result);
```

**Description**

The interface lrand48_r() shall function in the same way as the interface lrand48(), except that lrand48_r() shall use the data in `buffer` instead of the global random number generator state.

Before it is used, `buffer` must be initialized, for example, by calling lcong48_r(), seed48_r(), or srand48_r(), or by filling it with zeroes.

**memmem**

**Name**

memmem — locate bytes

**Synopsis**

```c
#define _GNU_SOURCE
#include <string.h>
void * memmem(const void * haystack, size_t haystacklen, const void * needle, size_t needlelen);
```

**Description**

memmem() finds the start of the first occurrence of the byte array referenced by `needle` of length `needlelen` in the memory area `haystack` of length `haystacklen`.

**Return Value**

If `needle` is found, memmem() returns a pointer to it. If `needlelen` is 0, memmem returns `haystack`. If `needle` is not found in `haystack`, memmem() returns NULL.

**Notes**

Earlier versions of the C library (prior to glibc 2.1) contained a memmem() with various problems, and application developers should treat this function with care.
memrchr

Name
memrchr — scan memory for a character

Synopsis
#include <string.h>
void * memrchr(const void * s, int c, size_t n);

Description
The memrchr() function shall locate the last occurrence of \( c \) (converted to an unsigned char) in the initial \( n \) bytes (each interpreted as an unsigned char) of the object pointed to by \( s \).

Return Value
The memrchr() shall return a pointer to the located byte, or a null pointer if the byte does not occur in the object.

Errors
No errors are defined.

See Also
memchr()

mkstemp64

Name
mkstemp64 — create a unique temporary file (Large File Support)

Synopsis
#include <stdio.h>
#include <stdlib.h>
int mkstemp64(char * template);

Description

`mkstemp64()` shall generates a unique temporary file name from `template`. The last six characters of `template` shall be `XXXXXX` and these are replaced with a string that makes the file name unique; the file is then created and an open file descriptor returned as described for `mkstemp()`.

`mkstemp64()` is a large-file version of the `mkstemp()` function as defined in POSIX 1003.1-2008 (ISO/IEC 9945-2009). The only difference is that the temporary file is opened with `open64()` instead of with `open()`.

Return Value

On success, `mkstemp64()` returns the file descriptor of the temporary file. Otherwise `mkstemp64()` shall return -1 and set `errno` to indicate the error.

Errors

See `mkstemp()` for possible error values.

mrand48_r

Name

`mrand48_r` — reentrantly generate pseudorandom numbers in a uniform distribution

Synopsis

```c
#include <stdlib.h>
int mrand48_r(struct drand48_data * buffer, long int * result);
```

Description

The interface `mrand48_r()` shall function in the same way as the interface `mrand48()`, except that `mrand48_r()` shall use the data in `buffer` instead of the global random number generator state.

Before it is used, `buffer` must be initialized, for example, by calling `lcong48_r()`, `seed48_r()`, or `srand48_r()`, or by filling it with zeroes.
mremap

Name
mremap — remap a virtual memory address

Synopsis
#include <sys/mman.h>
void * mremap(void * old_address, size_t old_size, size_t new_size, int flags);

Description
The mremap() function expands (or shrinks) an existing memory mapping, potentially moving it at the same time, depending on the flags argument and the available virtual address space.

old_address is the old address of the virtual memory block to be resized. Note that old_address must be page aligned. old_size is the old size of the virtual memory block. new_size is the requested size of the virtual memory block after the resize.

In Linux the memory is divided into pages. A user process has (one or) several linear virtual memory segments. Each virtual memory segment has one or more mappings to real memory pages (in the page table). Each virtual memory segment has its own protection (access rights), which may cause a segmentation violation if the memory is accessed incorrectly (e.g., writing to a read-only segment). Accessing virtual memory outside of the segments will also cause a segmentation violation.

mremap() uses the Linux page table scheme. mremap() changes the mapping between virtual addresses and memory pages. This can be used to implement a very efficient form of realloc().

The flags bit-mask argument may be 0, or include the following flag:

MREMAP_MAYMOVE
By default, if there is not sufficient space to expand a mapping at its current location, then mremap() fails. If this flag is specified, then the kernel is permitted to relocate the mapping to a new virtual address, if necessary. If the mapping is relocated, then absolute pointers into the old mapping location become invalid (offsets relative to the starting address of the mapping should be employed).

MREMAP_FIXED
This flag serves a similar purpose to the MAP_FIXED flag of mmap(). If this flag is specified, then mremap() accepts a fifth argument, void *new_address, which specifies a page aligned address to which the mapping must be moved. Any previous mapping at the address range specified by new_address and new_size is unmapped. If MREMAP_FIXED is specified, then MREMAP_MAYMOVE must also be specified.

If the memory segment specified by old_address and old_size is locked (using mlock() or similar), then this lock is maintained when the segment is resized and/or relocated. As a consequence, the amount of memory locked by the process may change.

Return Value
The mremap() function returns a pointer to the new virtual memory area on success. On error, the value MAP_FAILED is returned, and errno is set appropriately.

Errors
EAGAIN

The caller tried to expand a memory segment that is locked, but this was not possible without exceeding the RLIMIT_MEMLOCK resource limit.

EFAULT

"Segmentation fault." Some address in the range old_address to old_address+old_size is an invalid virtual memory address for this process. You can also get EFAULT even if there exist mappings that cover the whole address space requested, but those mappings are of different types.

EINVAL

An invalid argument was given. Possible causes are: old_address was not page aligned; a value other than MREMAP_MAYMOVE or MREMAP_FIXED was specified in flags; new_size was zero; new_size or new_address was invalid; or the new address range specified by new_address and new_size overlapped the old address range specified by old_address and old_size; or MREMAP_FIXED was specified without also specifying MREMAP_MAYMOVE.

ENOMEM

The memory area cannot be expanded at the current virtual address, and the MREMAP_MAYMOVE flag is not set in flags, or, there is not enough (virtual) memory available.
ngettext

Name

ngettext — search message catalogs for plural string

Synopsis

#include <libintl.h>
char * ngettext(const char * msgid1, const char * msgid2, unsigned long int n);

Description

The ngettext() function shall search the currently selected message catalogs for a string matching the singular string msgid1. If a string is located, and if n is 1, that string shall be returned. If n is not 1, a pluralized version (dependent on n) of the string shall be returned.

The ngettext() function is equivalent to dcngettext(NULL, msgid1, msgid2, n, LC_MESSAGES()).

Return Value

If a string is found in the currently selected message catalogs for msgid1, then if n is 1 a pointer to the located string shall be returned. If n is not 1, a pointer to an appropriately pluralized version of the string shall be returned. If no message could be found in the currently selected message catalogs, then if n is 1, a pointer to msgid1 shall be returned, otherwise a pointer to msgid2 shall be returned.

Applications shall not modify the string returned by ngettext().

Errors

None.

The ngettext() function shall not modify errno.

See Also

ggettext, dgettext, ngettext, dgettext, degettext, dcngettext, textdomain, bindtextdomain, bind_textdomain_codeset
nrand48_r

Name

nrand48_r — reentrantly generate pseudorandom numbers in a uniform distribution

Synopsis

#include <stdlib.h>
int nrand48_r(unsigned short[3] xsubi, struct drand48_data * buffer, long int * result);

Description

The interface nrand48_r() shall function in the same way as the interface nrand48(), except that nrand48_r() shall use the data in buffer instead of the global random number generator state.

Before it is used, buffer must be initialized, for example, by calling lcong48_r(), seed48_r(), or srand48_r(), or by filling it with zeroes.

openat64

Name

openat64 — open a file relative to a directory file descriptor (Large File Support)

Synopsis

#include <fcntl.h>
int openat64(int fd, const char * path, int oflag, ...);

Description

openat64() shall establish a connection between a file and a file descriptor. It shall be identical open64() except in the case where path specifies a relative path. In this case, the file to be opened shall be determined relative to the directory associated with the file descriptor fd instead of the current working directory.

openat64() is a large-file version of the openat() function as defined in POSIX.1003.1-2008 (ISO/IEC 9945-2009). It differs from openat() in the same way that open64() differs from open(), that the open is done in large-file mode.

Return Value

On success, openat64() returns a new file descriptor. Otherwise openat64() shall return -1 and set errno to indicate the error.

Errors

See openat() for possible error values.
pmap_getport

Name

pmap_getport — find the port number assigned to a service registered with a portmapper.

Synopsis

#include <rpc/pmap_clnt.h>

u_short * pmap_getport(struct sockaddr_in * address, const u_long program, const u_long * version, u_int protocol);

Description

The pmap_getport() function shall return the port number assigned to a service registered with a RPC Binding service running on a given target system, using the protocol described in RFC 1833: Binding Protocols for ONC RPC Version 2. The pmap_getport() function shall be called given the RPC program number program, the program version version, and transport protocol protocol. Conforming implementations shall support both IPPROTO_UDP and IPPROTO_TCP protocols. On entry, address shall specify the address of the system on which the portmapper to be contacted resides. The value of address->sin_port shall be ignored, and the standard value for the portmapper port shall always be used.

Note: Security and network restrictions may prevent a conforming application from contacting a remote RPC Binding Service.

Return Value

On success, the pmap_getport() function shall return the port number in host byte order of the RPC application registered with the remote portmapper. On failure, if either the program was not registered or the remote portmapper service could not be reached, the pmap_getport() function shall return 0. If the remote portmap service could not be reached, the status is left in the global variable rpc_createerr.

pmap_set

Name

pmap_set — establishes mapping to machine's RPC Bind service.

Synopsis

#include <rpc/pmap_clnt.h>

bool_t pmap_set(const u_long program, const u_long version, int protocol, u_short port);

Description

pmap_set() establishes a mapping between the triple [program, version, protocol] and port on the machine's RPC Bind service. The value of protocol is most likely IPPROTO_UDP or IPPROTO_TCP. Automatically done by svc_register().

Return Value

pmap_set() returns non-zero if it succeeds, 0 otherwise.
pmap_unset

**Name**
pmap_unset — destroys RPC Binding

**Synopsis**
```
#include <rpc/pmap_clnt.h>
bool_t pmap_unset(u_long prognum, u_long versnum);
```

**Description**
As a user interface to the RPC Bind service, pmap_unset() destroys all mapping between the triple [prognum, versnum, *] and ports on the machine's RPC Bind service.

**Return Value**
pmap_unset() returns non-zero if it succeeds, zero otherwise.

posix_fadvise64

**Name**
posix_fadvise64 — File advisory information (Large File Support)

**Synopsis**
```
#include <fcntl.h>
int posix_fadvise64(int fd, off64_t offset, off64_t len, int advice);
```

**Description**
The posix_fadvise64() function is a large-file version of the posix_fadvise() function defined in POSIX 1003.1-2008 (ISO/IEC 9945-2009). It shall advise the implementation on the expected behavior of the application with respect to the data in the file associated with the open file descriptor, fd, starting at offset and continuing for len bytes. The specified range need not currently exist in the file. If len is zero, all data following offset is specified. The implementation may use this information to optimize handling of the specified data. The posix_fadvise() function shall have no effect on the semantics of other operations on the specified data, although it may affect the performance of other operations.

The advice to be applied to the data is specified by the advice parameter, as specified in posix_fadvise().

**Return Value**
On success, posix_fadvise64() shall return 0. Otherwise an error number shall be returned to indicate the error. See posix_fadvise() for possible error values.
posix_fallocate64

**Name**
posix_fallocate64 — file space control (Large File Support)

**Synopsis**
```c
#include <fcntl.h>
int posix_fallocate64(int fd, off64_t offset, off64_t len);
```

**Description**
The `posix_fallocate64()` function is a large file version of `posix_fallocate()`. It shall behave as `posix_fallocate()` in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except that the `offset` and `len` arguments are `off64_t` objects rather than `off_t`.

**Return Value**
See `posix_fallocate()`.

**Errors**
See `posix_fallocate()`.

pread64

**Name**
pread64 — read from a file (Large File Support)

**Synopsis**
```c
#include <unistd.h>
ssize_t pread64(int fd, void *buf, size_t count, off64_t offset);
```

**Description**
pread64() shall read `count` bytes into `buf` from the file associated with the open file descriptor `fd`, at the position specified by `offset`, without changing the file position.

pread64() is a large-file version of the `pread()` function as defined in POSIX 1003.1-2008 (ISO/IEC 9945-2009). It differs from `pread()` in that the `offset` parameter is an `off64_t` instead of an `off_t`.

**Return Value**
On success, `pread64()` shall return the number of bytes actually read. Otherwise `pread64()` shall return -1 and set `errno` to indicate the error.

**Errors**
See `pread()` for possible error values.
ptrace

Name
ptrace — process trace

Synopsis
#include <sys/ptrace.h>
long ptrace(enum __ptrace_request request, pid_t pid, void * addr,
            void * data);

Description
The ptrace() system call shall enable a process to observe and control the execution
of another process, as well as examine and change certain attributes of that process.
This function operates via requests, which act on the traced process using the other para-
teters in ways unique to each request type. The tracing process must initiate tracing,
either via the PTRACE_TRACEME or PTRACE_ATTACH requests, before other requests
may be performed. Except for PTRACE_TRACEME and PTRACE_KILL, all requests must
be performed on a traced process that has been stopped.

All signals, except one, delivered to the traced process cause it to stop, irrespective of its
registered signal handling, and cause an event to be delivered to the tracing process
which can be detected using the wait(2) system call. The exception is the SIGKILL sig-
nal, which is delivered immediately and performs its usual specified behavior.

The following requests are defined:

PTRACE_TRACEME
This request initiates a trace from the perspective of the traced process, indicating
that the parent of the current process shall be the tracing process. When this is
called, a subsequent call to execve(2) shall cause the tracing process to receive a
SIGTRAP signal, and shall stop the current process. This is the only request a
traced process may perform, and a tracing process may not perform this request.
The other parameters are ignored.

PTRACE_ATTACH
This request initiates a trace from the perspective of the tracing process on the pro-
cess specified by pid. After this call succeeds, the traced process will appear to be
a child of the tracing process, although the original parent will still be returned to
the traced process via getppid(2). The traced process will receive a SIGSTOP sig-
nal; the tracing process should use wait(2) to ensure that the traced process has
stopped. A tracing process is only guaranteed to be able to trace its child processes;
the tracing of other processes may not be allowed by the system, and the process
with process ID 1 may not be traced under any circumstances. The addr and data
parameters are ignored.

PTRACE_CONT
This request restarts a traced process, given in pid, which has been stopped. The
data parameter may point to a signal ID to deliver to the traced process; if it is
zero or SIGSTOP, no signal is delivered to the child. The addr is ignored.

PTRACE_DETACH
This request performs the same function, in the same way, as PTRACE_CONT, ex-
cept that the tracing relationship between the tracing and traced processes is also
undone. If the trace was initiated using PTRACE_ATTACH, the original parent-child
relationships that existed beforehand are restored.
PTRACE_KILL

This request causes a SIGKILL signal to be sent to the traced process specified in \textit{pid}. The \textit{addr} and \textit{data} parameters are ignored.

PTRACE_PEEKTEXT

This request reads a word at the location \textit{addr} of the traced process \textit{pid}, and returns it to the caller. The \textit{data} parameter is ignored.

PTRACE_PEEKDATA

This request performs identically to the PTRACE_PEEKTEXT request.

PTRACE_PEEKUSER

This request reads a word at offset \textit{addr} in the USER area of the traced process \textit{pid}. The offset must be word-aligned. The \textit{data} parameter is ignored.

PTRACE_POKETEXT

This request writes the word pointed at by \textit{data} to the location \textit{addr} of the traced process \textit{pid}.

PTRACE_POKEDATA

This request performs identically to the PTRACE_POKETEXT request.

PTRACE_POKERUSER

This request writes the word pointed at by \textit{data} to offset \textit{addr} in the USER area of the traced process \textit{pid}. The offset must be word-aligned. Implementations may choose to disallow some modifications to the USER area.

PTRACE_GETREGS

This request copies the general purpose registers from the traced process \textit{pid} to the tracing process at location \textit{data}. This parameter may not be available on all architectures. The \textit{addr} parameter is ignored.

PTRACE_GETFPREGS

This request copies the floating point registers from the traced process \textit{pid} to the tracing process at location \textit{data}. This parameter may not be available on all architectures. The \textit{addr} parameter is ignored.

PTRACE_SETREGS

This request writes the general purpose registers to the traced process \textit{pid} from the tracing process at location \textit{data}. This parameter may not be available on all architectures. Implementations may choose to disallow some register modifications. The \textit{addr} parameter is ignored.

PTRACE_SETFPREGS

This request writes the floating point registers to the traced process \textit{pid} from the tracing process at location \textit{data}. This parameter may not be available on all architectures. Implementations may choose to disallow some register modifications. The \textit{addr} parameter is ignored.

PTRACE_GETSIGINFO

This request writes information about the signal which caused the traced process \textit{pid} to stop to the tracing process at location \textit{data}, as a siginfo_t. The \textit{addr} parameter is ignored.
PTRACE_SETSIGINFO

This request writes signal information to the traced process \texttt{pid} from a siginfo_t structure pointed at by \texttt{data}, such that it will be used as the signal information by the traced process when it is resumed. The \texttt{addr} parameter is ignored.

PTRACE_GETEVENTMSG

This request stores information about the most recent ptrace event for the traced process \texttt{pid} in the unsigned long pointed at by \texttt{data}. For PTRACE_EVENT_EXIT, this is the exit status of the traced process. For PTRACE_EVENT_FORK, PTRACE_EVENT_VFORK, or PTRACE_EVENT_CLONE, this is the PID of the newly created process. The \texttt{addr} parameter is ignored.

PTRACE_SYSCALL

This request performs the same function, in the same way, as PTRACE_CONT, but with the additional step of causing the traced process to stop at the next entry to or exit from a system call. The usual events that would also cause the traced process to stop continue to do so.

PTRACE_SINGLESTEP

This request performs the same function, in the same way, as PTRACE_CONT, but with the additional step of causing the traced process to stop after execution of a single instruction. The usual events that would also cause the traced process to stop continue to do so.

PTRACE_SYSEMU

This request performs the same function, in the same way, as PTRACE_CONT, but with the additional step of causing the traced process to stop on entry to the next syscall, which will then not be executed.

PTRACE_SYSEMU_SINGLESTEP

This request performs the same function, in the same way, as PTRACE_CONT, but with the additional step of causing the traced process to stop on entry to the next syscall, which will then not be executed. If the next instruction is not itself a syscall, the traced process will stop after a single instruction is executed.

PTRACE_SETOPTIONS

This request sets ptrace() options for the traced process \texttt{pid} from the location pointed to by \texttt{data}. The \texttt{addr} is ignored. This location is interpreted as a bitmask of options, as defined by the following flags:

PTRACE_O_TRACESYSGOOD

This option, when set, causes syscall traps to set bit 7 in the signal number.

PTRACE_O_TRACEFORK

This option, when set, causes the traced process to stop when it calls fork(2). The original traced process will stop with SIGTRAP | PTRACE_EVENT_FORK << 8, and the new process will be stopped with SIGSTOP. The new process will also be traced by the tracing process, as if the tracing process had sent the PTRACE_ATTACH request for that process. The PID of the new process may be retrieved with the PTRACE_GETEVENTMSG request.

PTRACE_O_TRACEVFORK
This option, when set, causes the traced process to stop when it calls vfork(2). The original traced process will stop with SIGTRAP | PTRACE_EVENT_VFORK << 8, and the new process will be stopped with SIGSTOP. The new process will also be traced by the tracing process, as if the tracing process had sent the PTRACE_ATTACH request for that process. The PID of the new process may be retrieved with the PTRACE_GETEVENTMSG request.

PTRACE_O_TRACECLONE

This option, when set, causes the traced process to stop when it calls clone(2). The original traced process will stop with SIGTRAP | PTRACE_EVENT_CLONE << 8, and the new process will be stopped with SIGSTOP. The new process will also be traced by the tracing process, as if the tracing process had sent the PTRACE_ATTACH request for that process. The PID of the new process may be retrieved with the PTRACE_GETEVENTMSG request. Under certain circumstances, clone(2) calls by the traced process will generate events and information consistent with the PTRACE_O_TRACEVFORK or PTRACE_O_TRACEFORK options above.

PTRACE_O_TRACEEXEC

This option, when set, causes the traced process to stop when it calls execve(2). The traced process will stop with SIGTRAP | PTRACE_EVENT_EXEC << 8.

PTRACE_O_TRACEVFORKDONE

This option, when set, causes the traced process to stop at the completion of its next vfork(2) call. The traced process will stop with SIGTRAP | PTRACE_EVENT_EXEC << 8.

PTRACE_O_TRACEEXIT

This option, when set, causes the traced process to stop upon exit. The traced process will stop with SIGTRAP | PTRACE_EVENT_EXIT << 8, and its exit status can be retrieved with the PTRACE_GETEVENTMSG request. The stop is guaranteed to be early in the process exit process, meaning that information such as register status at exit is preserved. Upon continuing, the traced process will immediately exit.

Return Value

On success, ptrace() shall return the requested data for PTRACE_PEEK requests, or zero for all other requests. On error, all requests return -1, with errno set to an appropriate value. Note that -1 may be a valid return value for PTRACE_PEEK requests; the application is responsible for distinguishing between an error condition and a valid return value in that case.

Errors

On error, ptrace() shall set errno to one of the regular error values below:

EBUSY

An error occurred while allocating or freeing a debug register.

EFAULT

The request attempted to read from or write to an invalid area in the memory space of the tracing or traced process.
EIO
The request was invalid, or it attempted to read from or write to an invalid area in
the memory space of the tracing or traced process, or it violated a word-alignment
boundary, or an invalid signal was given to continue the traced process.

EINVAL
An attempt was made to set an invalid option.

EPERM
The request to trace a process was denied by the system.

ESRCH
The process requested does not exist, is not being traced by the current process, or
is not stopped.

putwc_unlocked

Name
putwc_unlocked — non-thread-safe putwc

Description
putwc_unlocked() is the same as putwc(), except that it need not be thread-safe.
That is, it may only be invoked in the ways which are legal for getc_unlocked().

putwchar_unlocked

Name
putwchar_unlocked — non-thread-safe putwchar

Description
putwchar_unlocked() is the same as putwchar(), except that it need not be thread-
safe. That is, it may only be invoked in the ways which are legal for getc_unlocked().
pwrite64

**Name**
pwrite64 — write on a file (Large File Support)

**Synopsis**

```c
#include <unistd.h>
ssize_t pwrite64(int fd, const void *buf, size_t count, off64_t offset);
```

**Description**
pwrite64() shall write `count` bytes from `buf` to the file associated with the open file descriptor `fd`, at the position specified by `offset`, without changing the file position.

pwrite64() is a large-file version of the pwrite() function as defined in POSIX 1003.1-2008 (ISO/IEC 9945-2009). It differs from pwrite() in that the `offset` parameter is an off64_t instead of an off_t.

**Return Value**
On success, pwrite64() shall return the number of bytes actually written. Otherwise pwrite() shall return -1 and set `errno` to indicate the error.

**Errors**
See pwrite() for possible error values.

random_r

**Name**
random_r — reentrantly generate pseudorandom numbers in a uniform distribution

**Synopsis**

```c
#include <stdlib.h>
int random_r(struct random_data * buffer, int32_t * result);
```

**Description**
The interface random_r() shall function in the same way as the interface random(), except that random_r() shall use the data in `buffer` instead of the global random number generator state.

Before it is used, `buffer` must be initialized, for example, by calling lcong48_r(), seed48_r(), or srand48_r(), or by filling it with zeroes.
**readdir64_r**

**Name**

`readdir64_r` — read a directory (Large File Support)

**Synopsis**

```c
#include <dirent.h>
int readdir64_r(DIR * dirp, struct dirent64 * entry, struct dirent64 ** result);
```

**Description**

The `readdir64_r()` function is a large file version of `readdir_r()`. It shall behave as `readdir_r()` in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except that the `entry` and `result` arguments are `dirent64` structures rather than `dirent`.

**Return Value**

See `readdir_r()`.

**Errors**

See `readdir_r()`.

**regexec**

**Name**

`regexec` — regular expression matching

**Description**

The `regexec()` function shall behave as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except with differences as listed below.

**Differences**

Certain aspects of regular expression matching are optional; see Regular Expressions.

**scandir64**

**Name**

`scandir64` — scan a directory (Large File Support)

**Synopsis**

```c
#include <dirent.h>
int scandir64(const char * dir, const struct dirent64 ** namelist, int (* sel) (const struct dirent64 *), int (* compar) (const struct dirent64 **, const struct dirent64 **));
```

**Description**

`scandir64()` is a large-file version of the `scandir()` function as defined in POSIX 1003.1-2008 (ISO/IEC 9945-2009). If differs only in that the `namelist` and the parameters to the selection function `sel` and comparison function `compar` are of type `dirent64` instead of type `dirent`. 
scanf

Name

scanf — convert formatted input

Description

The scanf() family of functions shall behave as described in POSIX.1-2008
(ISO/IEC 9945-2009), except as noted below.

Differences

The %s, %S and %[ conversion specifiers shall accept an option length modifier a, which
shall cause a memory buffer to be allocated to hold the string converted. In such a case,
the argument corresponding to the conversion specifier should be a reference to a
pointer value that will receive a pointer to the allocated buffer. If there is insufficient
memory to allocate a buffer, the function may set errno to ENOMEM and a conversion
error results.

Note: This directly conflicts with the ISO C (1999) usage of %a as a conversion specifier for
hexadecimal float values. While this conversion specifier should be supported, a format spe-
cifier such as "%aseconds" will have a different meaning on an LSB conforming system.
sched_getaffinity

Name
sched_getaffinity — retrieve the affinity mask of a process

Synopsis
#include <sched.h>
int sched_getaffinity(pid_t pid, unsigned int cpusetsize, cpu_set_t *mask);

Description
sched_getaffinity() shall retrieve the affinity mask of a process.

The parameter pid specifies the ID for the process. If pid is 0, then the calling process
is specified instead.

The parameter cpusetsize specifies the length of the data pointed to by mask, in
bytes. Normally, this parameter is specified as sizeof(cpu_set_t).

Return Value
On success, sched_getaffinity() shall return 0, and the structure pointed to by
mask shall contain the affinity mask of the specified process.

On failure, sched_getaffinity() shall return -1 and set errno as follows.

Errors
EFAULT
Bad address.

EINVAL
 mask does not specify any processors that exist in the system, or cpusetsize is
smaller than the kernel's affinity mask.

ESRCH
The specified process could not be found.

See Also
sched_setscheduler(), sched_setaffinity().
sched_setaffinity

Name

sched_setaffinity — set the CPU affinity mask for a process

Synopsis

#include <sched.h>

int sched_setaffinity(pid_t pid, unsigned int cpusetsize, cpu_set_t *mask);

Description

sched_setaffinity() shall set the CPU affinity mask for a process.

The parameter `pid` specifies the ID for the process. If `pid` is 0, then the calling process is specified instead.

The parameter `cpusetsize` specifies the length of the data pointed to by `mask`, in bytes. Normally, this parameter is specified as `sizeof(cpu_set_t)`.

The parameter `mask` specifies the new value for the CPU affinity mask. The structure pointed to by `mask` represents the set of CPUs on which the process may run. If `mask` does not specify one of the CPUs on which the specified process is currently running, then sched_setaffinity() shall migrate the process to one of those CPUs.

Setting the mask on a multiprocessor system can improve performance. For example, setting the mask for one process to specify a particular CPU, and then setting the mask of all other processes to exclude the CPU, dedicates the CPU to the process so that the process runs as fast as possible. This technique also prevents loss of performance in case the process terminates on one CPU and starts again on another, invalidating cache.

Return Value

On success, sched_setaffinity() shall return 0.

On failure, sched_setaffinity() shall return -1 and set `errno` as follows.

Errors

EFAULT

Bad address.

EINVAL

`mask` does not specify any processors that exist in the system, or `cpusetsize` is smaller than the kernel's affinity mask.

EPERM

Insufficient privileges. The effective user ID of the process calling sched_setaffinity() is not equal to the user ID or effective user ID of the specified process, and the calling process does not have appropriate privileges.

ESRCH

The specified process could not be found.

See Also

sched_setscheduler(), sched_getaffinity().
**sched_setscheduler**

**Name**

`sched_setscheduler` — set scheduling policy and parameters

**Synopsis**

```c
#include <sched.h>
int sched_setscheduler(pid_t pid, int policy, const struct sched_param *param);
```

**Description**

The `sched_setscheduler()` shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except as noted below.

**Return Value**

On success, 0 is returned instead of the former scheduling policy.

**seed48_r**

**Name**

`seed48_r` — reentrantly generate pseudorandom numbers in a uniform distribution

**Synopsis**

```c
#include <stdlib.h>
int seed48_r(unsigned short[3] seed16v, struct drand48_data *buffer);
```

**Description**

The interface `seed48_r()` shall function in the same way as the interface `seed48()`, except that `seed48_r()` shall use the data in `buffer` instead of the global random number generator state.
sendfile

Name
sendfile — transfer data between two file descriptors

Synopsis
#include <sys/sendfile.h>
ssize_t sendfile(int out_fd, int in_fd, off_t * offset, size_t count);

Description
The sendfile() function shall copy data between the file descriptor in_fd, which
must not be a socket, and the file descriptor out_fd, which must be a socket. in_fd
should be opened for reading, and out_fd should be opened for writing.

The offset parameter points to a variable set to the file offset at which sendfile()
shall start reading from in_fd, unless it is NULL. On exit, this variable shall contain the
offset of the byte immediately after the last byte read. sendfile() shall not change the
current file offset of in_fd, unless it is NULL. In that case, sendfile() shall adjust the
current file offset to show how many bytes were read.

The count parameter specifies how many bytes to copy.

Return Value
On success, sendfile() shall return the number of bytes written to out_fd.
On failure, sendfile() shall return -1 and set errno appropriately, as follows.

Errors
EAGAIN
Non-blocking I/O with O_NONBLOCK has been chosen, but the write would block.

EBADF
The input file is not open for reading, or the output file is not open for writing.

EFAULT
Bad address.

EINVAL
An mmap()-like operation is unavailable for in_fd, or file descriptor is locked or
invalid.

EIO
There was an unspecified error while reading.

ENOMEM
There is not enough memory to read from in_fd.

Notes
sendfile() is usually faster than combining read() and write() calls, because it is
part of the kernel. However, if it fails with EINVAL, falling back to read() and
write() may be advisable.
It is advisable for performance reasons to use the TCP_CORK option of the tcp() function when sending header data with file contents to a TCP socket. This minimizes the number of packets.

See Also
mmap(), open(), socket(), splice().

sendfile64

Name
sendfile64 — transfer data between two file descriptors (Large File Support)

Synopsis

```c
#include <sys/sendfile.h>
ssize_t sendfile64(int out_fd, int in_fd, off64_t * offset, size_t count);
```

Description
The sendfile64() function is a large-file version of the sendfile() function.

setbuffer

Name
setbuffer — stream buffering operation

Synopsis

```c
#include <stdio.h>
void setbuffer(FILE * stream, char * buf, size_t size);
```

Description
setbuffer() is an alias for the call to setvbuf(). It works the same, except that the size of the buffer in setbuffer() is up to the caller, rather than being determined by the default BUFSIZ.
setgroups

Name
setgroups — set list of supplementary group IDs

Synopsis
#include <grp.h>
int setgroups(size_t size, const gid_t * list);

Description
If the process has appropriate privilege, the `setgroups()` function shall set the supplementary group IDs for the current process. `list` shall reference an array of `size` group IDs. A process may have at most `NGROUPS_MAX` supplementary group IDs.

Return Value
On successful completion, 0 is returned. On error, -1 is returned and the `errno` is set to indicate the error.

Errors
EFAULT
`list` has an invalid address.

EPERM
The process does not have appropriate privileges.

EINVAL
`size` is greater than `NGROUPS_MAX`.

sethostname

Name
sethostname — set host name

Synopsis
#include <unistd.h>
#include <sys/param.h>

© 2015 Linux Foundation

#include <sys/utsname.h>

int sethostname(const char * name, size_t len);

Description

If the process has appropriate privileges, the sethostname() function shall change the host name for the current machine. The name shall point to a null-terminated string of at most len bytes that holds the new hostname.

If the symbol HOST_NAME_MAX is defined, or if sysconf(_SC_HOST_NAME_MAX)() returns a value greater than 0, this value shall represent the maximum length of the new hostname. Otherwise, if the symbol MAXHOSTLEN is defined, this value shall represent the maximum length for the new hostname. If none of these values are defined, the maximum length shall be the size of the nodename field of the utsname structure.

Return Value

On success, 0 is returned. On error, -1 is returned and the global variable errno is set appropriately.

Errors

EINVAL

len is negative or larger than the maximum allowed size.

EPERM

the process did not have appropriate privilege.

EFAULT

name is an invalid address.

Rationale

POSIX 1003.1-2008 (ISO/IEC 9945-2009) guarantees that:

Maximum length of a host name (not including the terminating null) as returned from the gethostname() function shall be at least 255 bytes.

The glibc C library does not currently define HOST_NAME_MAX, and although it provides the name _SC_HOST_NAME_MAX a call to sysconf() returns -1 and does not alter errno in this case (indicating that there is no restriction on the hostname length). However, the glibc manual indicates that some implementations may have MAXHOSTNAMELEN as a means of detecting the maximum length, while the Linux kernel at release 2.4 and 2.6 stores this hostname in the utsname structure. While the glibc manual suggests simply shortening the name until sethostname() succeeds, the LSB requires that one of the first four mechanisms works. Future versions of glibc may provide a more reasonable result from sysconf(_SC_HOST_NAME_MAX).

setsockopt

Name

setsockopt — set socket options

Synopsis

#include <sys/socket.h>
```c
#include <netinet/ip.h>
int setsockopt(int socket, int level, int option_name, const void *option_value, socklen_t option_len);
```

**Description**

The `setsockopt()` function shall behave as specified in `POSIX.1003.1-2008`\n(ISO/IEC 9945-2009), with the following extensions.

**IP Protocol Level Options**

If the `level` parameter is `IPPROTO_IP`, the following values shall be supported for `option_name` (see `RFC 791:Internet Protocol` for further details):

**IP_OPTIONS**

Set the Internet Protocol options sent with every packet from this socket. The `option_value` shall point to a memory buffer containing the options and `option_len` shall contain the size in bytes of that buffer. For IPv4, the maximum length of options is 40 bytes.

**IP_TOS**

Set the Type of Service flags to use when sending packets with this socket. The `option_value` shall point to a value containing the type of service value. The least significant two bits of the value shall contain the new Type of Service indicator. Use of other bits in the value is unspecified. The `option_len` parameter shall hold the size, in bytes, of the buffer referred to by `option_value`.

**IP_TTL**

Set the current unicast Internet Protocol Time To Live value used when sending packets with this socket. The `option_value` shall point to a value containing the time to live value, which shall be between 1 and 255. The `option_len` parameter shall hold the size, in bytes, of the buffer referred to by `option_value`.

**IP_MULTICAST_TTL**

Sets the Time To Live value of outgoing multicast packets for this socket. `optval` shall point to an integer which contains the new TTL value. If the new TTL value is -1, the implementation should use an unspecified default TTL value. If the new TTL value is out of the range of acceptable values (0-255), `setsockopt()` shall return -1 and set `errno` to indicate the error.

**IP_MULTICAST_LOOP**

Sets a boolean flag indicating whether multicast packets originating locally should be looped back to the local sockets. `optval` shall point to an integer which contains the new flag value.

**IP_ADD_MEMBERSHIP**

Join a multicast group. `optval` shall point to a `ip_mreq` structure. Before calling, the caller should fill in the `imr_multiaddr` field with the multicast group address and the `imr_address` field with the address of the local interface. If `imr_address` is set to `INADDR_ANY`, then an appropriate interface is chosen by the system.

**IP_DROP_MEMBERSHIP**

Leave a multicast group. `optval` shall point to a `ip_mreq` structure containing the same values as were used with `IP_ADD_MEMBERSHIP`.

**IP_MULTICAST_IF**
Set the local device for a multicast socket. `optval` shall point to either an `ip_mreqn` structure or an `in_addr` structure. If using the `ip_mreqn` structure, the `imr_multiaddr` field should be set to multicast group address, the `imr_address` field to the address of the local interface, and the `imr_index` field to the interface index. If using the `in_addr` structure, the address of the local interface shall be specified. If `in_addr` or `imr_address` is set to `INADDR_ANY`, then an appropriate interface is chosen by the system. If `imr_index` is zero, then an appropriate interface index is chosen by the implementation.

The `ip_mreq` structure contains two `struct in_addr` fields: `imr_multiaddr` and `imr_address`.

### Return Value

On success, 0 is returned. On error, -1 is returned and the global variable `errno` is set appropriately.

### Errors


### setstate_r

#### Name

`setstate_r` — reentrantly change the state array used by random number generator functions

#### Synopsis

```c
#include <stdlib.h>
int setstate_r(char * statebuf, struct random_data * buf);
```

#### Description

The interface `setstate_r()` shall function in the same way as the interface `setstate()`, except that `setstate_r()` shall use the data in `statebuf` instead of the global random number generator state.

### setutent

#### Name

`setutent` — access user accounting database entries

#### Synopsis

```c
#include <utmp.h>
void setutent(void);
```

#### Description

The `setutent()` function shall reset the user accounting database such that the next call to `getutent()` shall return the first record in the database. It is recommended to call it before any of the other functions that operate on the user accounting databases (e.g. `getutent()`)

#### Return Value

None.
sigandset

Name

sigandset — build a new signal set by combining the two input sets using logical AND

Synopsis

#include <signal.h>
int sigandset(sigset_t * set, const sigset_t * left, const sigset_t * right);

Description

The sigandset() function shall combine the two signal sets referenced by left and right, using a logical AND operation, and shall place the result in the location referenced by set. The resulting signal set shall contain only signals that are in both the set referenced by left and the set referenced by right.

Applications shall call sigemptyset() or sigfillset() at least once for each object of type sigset_t to initialize it. If an uninitialized or NULL object is passed to sigandset(), the results are undefined.

Return Value

sigandset() returns 0. There are no defined error returns.

See Also

sigorset()

sigisemptyset

Name

sigisemptyset — check for empty signal set

Synopsis

#include <signal.h>
int sigisemptyset(const sigset_t * set);

Description

The sigisemptyset() function shall check for empty signal set referenced by set.

Applications shall call sigemptyset() or sigfillset() at least once for each object of type sigset_t to initialize it. If an uninitialized or NULL object is passed to sigisemptyset(), the results are undefined.

Return Value

The sigisemptyset() function shall return a positive non-zero value if the signal set referenced by set is empty, or zero if this set is empty. There are no defined error returns.
sigorset

Name

sigorset — build a new signal set by combining the two input sets using logical OR

Synopsis

#include <signal.h>
int sigorset(sigset_t * set, const sigset_t * left, const sigset_t * right);

Description

The sigorset() function shall combine the two signal sets referenced by left and right, using a logical OR operation, and shall place the result in the location referenced by set. The resulting signal set shall contain only signals that are in either the set referenced by left or the set referenced by right.

Applications shall call sigemptyset() or sigfillset() at least once for each object of type sigset_t to initialize it. If an uninitialized or NULL object is passed to sigorset(), the results are undefined.

Return Value

sigorset() returns 0. There are no defined error returns.

See Also

sigandset()

sigpause

Name

sigpause — remove a signal from the signal mask and suspend the thread (deprecated)

Synopsis

#include <signal.h>
int sigpause(int sig);

Description

The sigpause() function is deprecated from the LSB and is expected to disappear from a future version of the LSB. Conforming applications should use sigsuspend() instead.

In the source standard, sigpause() is implemented as a macro causing it to behave as described in POSIX.1003.1-2008 (ISO/IEC 9945-2009), and is equivalent to the function __xpg_sigpause(). If the macro is undefined, sigpause() from the binary standard is used, with differences as described here:

The sigpause() function shall block those signals indicated by sig and suspend execution of the thread until a signal is delivered. When a signal is delivered, the original signal mask shall be restored.

See Also

__xpg_sigpause()
sigreturn

Name
sigreturn — return from signal handler and cleanup stack frame

Synopsis
int sigreturn(struct sigcontext * scp);

Description
The sigreturn() function is used by the system to cleanup after a signal handler has returned. This function is not in the source standard; it is only in the binary standard.

Return Value
sigreturn() never returns.

srand48_r

Name
srand48_r — reentranently generate pseudorandom numbers in a uniform distribution

Synopsis
#include <stdlib.h>
int srand48_r(long int seedval, struct drand48_data * buffer);

Description
The interface srand48_r() shall function in the same way as the interface srand48(), except that srand48_r() shall use the data in buffer instead of the global random number generator state.

srandom_r

Name
srandom_r — reentranently set the seed for a new sequence of pseudorandom numbers

Synopsis
#include <stdlib.h>
int srandom_r(unsigned int seed, struct random_data * buffer);

Description
The interface srandom_r() shall function in the same way as the interface srandom(), except that srandom_r() shall use the data in buffer instead of the global random number generator state.
**sscanf**

**Name**

*sscanf* — convert formatted input

**Description**

The `scanf()` family of functions shall behave as described in *POSIX.1-2008* (ISO/IEC 9945-2009), except as noted below.

**Differences**

The `%s`, `%S` and `[%` conversion specifiers shall accept an option length modifier `a`, which shall cause a memory buffer to be allocated to hold the string converted. In such a case, the argument corresponding to the conversion specifier should be a reference to a pointer value that will receive a pointer to the allocated buffer. If there is insufficient memory to allocate a buffer, the function may set `errno` to `ENOMEM` and a conversion error results.

**Note:** This directly conflicts with the *ISO C (1999)* usage of `%a` as a conversion specifier for hexadecimal float values. While this conversion specifier should be supported, a format specifier such as `"%aseconds"` will have a different meaning on an LSB conforming system.
**statfs**

**Name**

**statfs** — (deprecated)

**Synopsis**

```
#include <sys/statfs.h>
int statfs(const char *path, struct statfs *buf);
```

**Description**

The **statfs()** function returns information about a mounted file system. The file system is identified by *path*, a path name of a file within the mounted filesystem. The results are placed in the structure pointed to by *buf*.

Fields that are undefined for a particular file system shall be set to 0.

**Note:** Application developers should use the **statvfs()** function to obtain general file system information. Applications should only use the **statfs()** function if they must determine the file system type, which need not be provided by **statvfs()**.

**Return Value**

On success, the **statfs()** function shall return 0 and set the fields of the structure identified by *buf* accordingly. On error, the **statfs()** function shall return -1 and set *errno* accordingly.

**Errors**

- **ENOTDIR**
  
  A component of the path prefix of *path* is not a directory.

- **ENAMETOOLONG**
  
  *path* is too long.

- **ENOENT**
  
  The file referred to by *path* does not exist.

- **EACCES**
  
  Search permission is denied for a component of the path prefix of *path*.

- **ELOOP**
  
  Too many symbolic links were encountered in translating *path*.

- **EFAULT**
  
  *buf* or *path* points to an invalid address.

- **EIO**
  
  An I/O error occurred while reading from or writing to the file system.

- **ENOMEM**
  
  Insufficient kernel memory was available.

- **ENOSYS**
The filesystem path is on does not support statfs().

statfs64

Name
statfs64 — (deprecated)

Synopsis
#include <sys/statfs.h>
int statfs64(const char * path, struct statfs64 *buf);

Description
The statfs64() function returns information about a mounted file system. The file system is identified by path, a path name of a file within the mounted filesystem. The results are placed in the structure pointed to by buf.

statfs64() is a large-file version of the statfs() function.

Fields that are undefined for a particular file system shall be set to 0.

Note: Application developers should use the statvfs64() function to obtain general file system information. Applications should only use the statfs64() function if they must determine the file system type, which need not be provided by statvfs64().

Return Value
On success, the statfs64() function shall return 0 and set the fields of the structure identified by buf accordingly. On error, the statfs64() function shall return -1 and set errno accordingly.

Errors
See fstatfs().

stime

Name
stime — set time

Synopsis
#define _SVID_SOURCE
#include <time.h>
int stime(const time_t * t);

Description
If the process has appropriate privilege, the stime() function shall set the system's idea of the time and date. Time, referenced by t, is measured in seconds from the epoch (defined in POSIX 1003.1-2008 (ISO/IEC 9945-2009) as 00:00:00 UTC January 1, 1970).

Return Value
On success, stime() shall return 0. Otherwise, stime() shall return -1 and errno shall be set to indicate the error.

Errors
EPERM
The process does not have appropriate privilege.
EINVAL
t is a null pointer.

strcasestr
Name
strcasestr — locate a substring ignoring case

Synopsis
#include <string.h>
char * strcasestr(const char * s1, const char * s2);

Description
The strcasestr() shall behave as strstr(), except that it shall ignore the case of both strings. The strcasestr() function shall be locale aware; that is strcasestr() shall behave as if both strings had been converted to lower case in the current locale before the comparison is performed.

Return Value
Upon successful completion, strcasestr() shall return a pointer to the located string or a null pointer if the string is not found. If s2 points to a string with zero length, the function shall return s1.
`strerror_r`

**Name**

`strerror_r` — return string describing error number

**Synopsis**

```c
#include <string.h>
char * strerror_r(int errnum, char * buf, size_t buflen);
```

**Description**

In the source standard, `strerror_r()` is implemented as a macro causing it to behave as described in POSIX_1003.1-2008 (ISO/IEC 9945-2009), and is equivalent to the function `__xpg_strerror_r()`. If the macro is undefined, `strerror_r()` from the binary standard is used, with differences as described here.

The `strerror_r()` function shall return a pointer to the string corresponding to the error number `errno`. The returned pointer may point within the buffer `buf` (at most `buflen` bytes).

**Return Value**

On success, `strerror_r()` shall return a pointer to the generated message string (determined by the setting of the LC_MESSAGES category in the current locale). Otherwise, `strerror_r()` shall return the string corresponding to "Unknown error".

**See Also**

`__xpg_strerror_r()`
strptime

Name
strptime — parse a time string

Description
The strptime() shall behave as specified in the POSIX 1003.1-2008 (ISO/IEC 9945-2009) with differences as listed below.

Number of leading zeroes may be limited
The POSIX 1003.1-2008 (ISO/IEC 9945-2009) specifies fields for which "leading zeros are permitted but not required"; however, applications shall not expect to be able to supply more leading zeroes for these fields than would be implied by the range of the field. Implementations may choose to either match an input with excess leading zeroes, or treat this as a non-matching input. For example, %j has a range of 001 to 366, so 0, 00, 000, 001, and 045 are acceptable inputs, but inputs such as 0000, 0366 and the like are not.

Rationale
glibc developers consider it appropriate behavior to forbid excess leading zeroes. When trying to parse a given input against several format strings, forbidding excess leading zeroes could be helpful. For example, if one matches 011-12-26 against %m-%d-%Y and then against %Y-%m-%d, it seems useful for the first match to fail, as it would be perverse to parse that date as November 12, year 26. The second pattern parses it as December 26, year 11.

The POSIX 1003.1-2008 (ISO/IEC 9945-2009) is not explicit that an unlimited number of leading zeroes are required, although it may imply this. The LSB explicitly allows implementations to have either behavior. Future versions of this standard may require implementations to forbid excess leading zeroes.

An Interpretation Request is currently pending against POSIX 1003.1-2008 (ISO/IEC 9945-2009) for this matter.
strsep

Name
strsep — extract token from string

Synopsis

```
#include <string.h>
char * strsep(char ** stringp, const char * delim);
```

Description

The `strsep()` function shall find the first token in the string referenced by the pointer `stringp`, using the characters in `delim` as delimiters.

If `stringp` is NULL, `strsep()` shall return NULL and do nothing else.

If `stringp` is non-NULL, `strsep()` shall find the first token in the string referenced by `stringp`, where tokens are delimited by characters in the string `delim`. This token shall be terminated with a \0 character by overwriteing the delimiter, and `stringp` shall be updated to point past the token. In case no delimiter was found, the token is taken to be the entire string referenced by `stringp`, and the location referenced by `stringp` is made NULL.

Return Value

`strsep()` shall return a pointer to the beginning of the token.

Notes

The `strsep()` function was introduced as a replacement for `strtok()`, since the latter cannot handle empty fields. However, `strtok()` conforms to ISO C (1999) and to POSIX 1003.1-2008 (ISO/IEC 9945-2009) and hence is more portable.

See Also

`strtok()`, `strtok_r()`.

strtoq

Name

strtoq — convert string value to a long or quad_t integer

Synopsis

```
#include <sys/types.h>
#include <stdlib.h>
```
#include <limits.h>
long long strtolq(const char * nptr, char ** endptr, int base);

**Description**

`strtolq()` converts the string `nptr` to a quad value. The conversion is done according to the given base, which shall be between 2 and 36 inclusive, or be the special value 0. `nptr` may begin with an arbitrary amount of white space (as determined by `isspace()`), followed by a single optional + or - sign character. If `base` is 0 or 16, the string may then include a 0x prefix, and the number will be read in base 16; otherwise, a 0 base is taken as 10 (decimal), unless the next character is 0, in which case it is taken as 8 (octal).

The remainder of the string is converted to a long value in the obvious manner, stopping at the first character which is not a valid digit in the given base. (In bases above 10, the letter A in either upper or lower case represents 10, B represents 11, and so forth, with Z representing 35.)

**Return Value**

`strtolq()` returns the result of the conversion, unless the value would underflow or overflow. If an underflow occurs, `strtolq()` returns QUAD_MIN. If an overflow occurs, `strtolq()` returns QUAD_MAX. In both cases, the global variable `errno` is set to ERANGE.

**Errors**

ERANGE

The given string was out of range; the value converted has been clamped.

**Synopsis**

```
#include <sys/types.h>
#include <stdlib.h>
```
`#include <limits.h>`

```c
unsigned long long strtouq(const char * nptr, char ** endptr, int base);
```

### Description

`strtouq()` converts the string `nptr` to an unsigned long long value. The conversion is done according to the given base, which shall be between 2 and 36 inclusive, or be the special value 0.

`nptr` may begin with an arbitrary amount of white space (as determined by `isspace()`), followed by a single optional + or - sign character. If `base` is 0 or 16, the string may then include a 0x prefix, and the number will be read in base 16; otherwise, a 0 base is taken as 10 (decimal), unless the next character is 0, in which case it is taken as 8 (octal).

The remainder of the string is converted to an unsigned long value in the obvious manner, stopping at the end of the string or at the first character that does not produce a valid digit in the given base. (In bases above 10, the letter A in either upper or lower case represents 10, B represents 11, and so forth, with Z representing 35.)

### Return Value

On success, `strtouq()` returns either the result of the conversion or, if there was a leading minus sign, the negation of the result of the conversion, unless the original (non-negated) value would overflow. In the case of an overflow the function returns `UQUAD_MAX` and the global variable `errno` is set to `ERANGE`.

### Errors

`ERANGE`

The given string was out of range; the value converted has been clamped.

---

### svc_register

```c
bool_t svc_register(SVCXPRT * xprt, rpcprog_t prognum, rpcvers_t versnum, __dispatch_fn_t dispatch, rpcprot_t protocol);
```

### Description

The `svc_register()` function shall associate the program identified by `prognum` at version `versnum` with the service dispatch procedure, `dispatch`. If `protocol` is zero, the service is not registered with the portmap service. If `protocol` is non-zero, then a mapping of the triple `[prognum, versnum, protocol]` to `xprt->xp_port` is established with the local portmap service. The procedure `dispatch` has the following form:

```c
int dispatch(struct svc_req * request, SVCXPRT * xprt);
```

### Return Value

`svc_register()` returns 1 if it succeeds, and zero otherwise.
**svc_run**

**Name**

svc_run — waits for RPC requests to arrive and calls service procedure

**Synopsis**

```c
#include <rpc/svc.h>
void svc_run(void);
```

**Description**

The svc_run() function shall wait for RPC requests to arrive, read and unpack each request, and dispatch it to the appropriate registered handler. Under normal conditions, svc_run() shall not return; it shall only return if serious errors occur that prevent further processing.

**svc_sendreply**

**Name**

svc_sendreply — called by RPC service's dispatch routine

**Synopsis**

```c
bool_t svc_sendreply(SVCXPRT *xprt, xdrproc_t outproc, caddr_t out);
```

**Description**

Called by an RPC service's dispatch routine to send the results of a remote procedure call. The parameter xprt is the request's associated transport handle; outproc is the XDR routine which is used to encode the results; and out is the address of the results. This routine returns one if it succeeds, zero otherwise.
svctcp_create

Name
svctcp_create — create a TCP/IP-based RPC service transport

Synopsis
#include <rpc/rpc.h>
SVCXPRT * svctcp_create(int sock, u_int send_buf_size, u_int recv_buf_size);

Description
svctcp_create() creates a TCP/IP-based RPC service transport, to which it returns a pointer. The transport is associated with the socket sock, which may be RPC_ANYSOCK, in which case a new socket is created. If the socket is not bound to a local TCP port, then this routine binds it to an arbitrary port. Upon completion, xprt->xp_sock is the transport's socket descriptor, and xprt->xp_port is the transport's port number. Since TCP-based RPC uses buffered I/O, users may specify the size of buffers; values of zero choose suitable defaults.

Return Value
svctcp_create() returns NULL if it fails, or a pointer to the RPC service transport otherwise.

svcup_create

Name
svcup_create — create a UDP-based RPC service transport

Synopsis
SVCXPRT *
svcup_create(int sock);

Description
The svcup_create() function shall create a UDP/IP-based RPC service transport, and return a pointer to its descriptor. The transport is associated with the socket sock, which may be RPC_ANYSOCK, in which case a new socket shall be created. If the socket is not bound to a local UDP port, then svcup_create() shall bind it to an arbitrary port.

If svcup_create() returns successfully, then the xp_sock field in the result shall be the transport's socket descriptor, and the xp_port field shall be the transport's port number.

Return Value
Upon successful completion, svcup_create() shall return a pointer to a RPC service transport; otherwise, a null pointer shall be returned.
swscanf

Name
swscanf — convert formatted input

Description
The scanf() family of functions shall behave as described in POSIX.1-2008 (ISO/IEC 9945-2009), except as noted below.

Differences
The %s, %S and %[ conversion specifiers shall accept an option length modifier a, which shall cause a memory buffer to be allocated to hold the string converted. In such a case, the argument corresponding to the conversion specifier should be a reference to a pointer value that will receive a pointer to the allocated buffer. If there is insufficient memory to allocate a buffer, the function may set errno to ENOMEM and a conversion error results.

Note: This directly conflicts with the ISO C (1999) usage of %a as a conversion specifier for hexadecimal float values. While this conversion specifier should be supported, a format specifier such as "%aseconds" will have a different meaning on an LSB conforming system.
sysconf

**Name**

*sysconf* — Get configuration information at runtime

**Synopsis**

```c
#include <unistd.h>
long sysconf(int name);
```

**DESCRIPTION**

`sysconf()` is as specified in [POSIX.1-2008 (ISO/IEC 9945-2009)](https://www.opengroup.org/onlinepubs/9699919999/), but with differences as listed below.

**Extra Variables**

These additional values extend the list in [POSIX.1-2008 (ISO/IEC 9945-2009)].

- `_SC_PHYS_PAGES`
  The number of pages of physical memory.

- `_SC_AVPHYS_PAGES`
  The number of currently available pages of physical memory.

- `_SC_NPROCESSORS_CONF`
  The number of processors configured.

- `_SC_NPROCESSORS_ONLN`
  The number of processors currently online (available).

**Extra Versions**

While this specification only requires conformance with [POSIX.1-2008 (ISO/IEC 9945-2009)](https://www.opengroup.org/onlinepubs/9699919999/), implementations are not constrained from moving on and claiming conformance with a subsequent edition, [POSIX.1-2008 (ISO/IEC 9945-2009)]. Thus for run-time checks using `sysconf()`, the wording is amended to allow return values of 0, -1, 200112L or 200809L where formerly 200809L was not listed as allowed.
sysinfo

Name
sysinfo — return system information

Synopsis
#include <sys/sysinfo.h>
int sysinfo(struct sysinfo *info);

Description
sysinfo() provides a way to obtain certain system statistics. Statistics are written into a sysinfo structure pointed to by info. Elements which take a size are sized in units indicated by the value of the mem_unit member of info. The other members have traditional meanings as indicated in Data Definitions, but are not formally part of this specification.

Return Value
Returns zero on success. On error, -1 is returned and errno is set to indicate the error.

Errors
EFAULT
The info parameter does not point to a valid sysinfo structure.
system

Name
system — execute a shell command

Synopsis
#include <stdlib.h>
int system(const char * string);

Description
The system() function shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009).

Notes
The fact that system() ignores interrupts is often not what a program wants. POSIX 1003.1-2008 (ISO/IEC 9945-2009) describes some of the consequences; an additional consequence is that a program calling system() from a loop cannot be reliably interrupted. Many programs will want to use the exec() family of functions instead.

Do not use system() from a program with suid or sgid privileges, because unexpected values for some environment variables might be used to subvert system integrity. Use the exec() family of functions instead, but not execvp() or execvp(). system() will not, in fact, work properly from programs with suid or sgid privileges on systems on which /bin/sh is bash version 2, since bash 2 drops privileges on startup. (Debian uses a modified bash which does not do this when invoked as sh.)

The check for the availability of /bin/sh is not actually performed; it is always assumed to be available. ISO C (1999) specifies the check, but POSIX 1003.1-2008 (ISO/IEC 9945-2009) specifies that the return shall always be nonzero, since a system without the shell is not conforming, and it is this that is implemented.

It is possible for the shell command to return 127, so that code is not a sure indication that the execve() call failed; check the global variable errno to make sure.
textdomain

Name
textdomain — set the current default message domain

Synopsis
#include <libintl.h>
char * textdomain(const char * domainname);

Description
The textdomain() function shall set the current default message domain to domainname. Subsequent calls to gettext() and ngettext() use the default message domain.

If domainname is NULL, the default message domain shall not be altered.

If domainname is "", textdomain() shall reset the default domain to the system default of "messages".

Return
On success, textdomain() shall return the currently selected domain. Otherwise, a null pointer shall be returned, and errno is set to indicate the error.

Errors
ENOMEM
Insufficient memory available.

unlink

Name
unlink — remove a directory entry

Synopsis
int unlink(const char * path);

Description
unlink() is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

See also Section 18.1, Additional behaviors: unlink/link on directory.

May return EISDIR on directories

If path specifies a directory, the implementation may return EISDIR instead of EPERM as specified by POSIX 1003.1-2008 (ISO/IEC 9945-2009).

Rationale: The Linux kernel has deliberately chosen EISDIR for this case and does not expect to change.
utmpname

Name
utmpname — set user accounting database

Synopsis
#include <utmp.h>
int utmpname(const char * dbname);

Description
The utmpname() function shall cause the user accounting database used by the getutent(), getutent_r(), getutxent(), getutxid(), getutxline(), and pututxline() functions to be that named by dbname, instead of the system default database. See Section 18.3 for further information.

Note: The LSB does not specify the format of the user accounting database, nor the names of the file or files that may contain it.

Return Value
None.

Errors
None defined.

vasprintf

Name
vasprintf — write formatted output to a dynamically allocated string

Synopsis
#include <stdarg.h>
#include <stdio.h>
int vasprintf(char ** restrict ptr, const char * restrict format, va_list arg);

Description
The vasprintf() function shall write formatted output to a dynamically allocated string, and store the address of that string in the location referenced by ptr. It shall behave as asprintf(), except that instead of being called with a variable number of arguments, it is called with an argument list as defined by <stdarg.h>.

Return Value
Refer to fprintf().

Errors
Refer to fprintf().
**verrx**

**Name**

*verrx* — display formatted error message and exit

**Synopsis**

```c
#include <stdarg.h>
#include <err.h>
void verrx (int eval, const char *fmt, va_list args);
```

**Description**

The *verrx()* shall behave as *errx()* except that instead of being called with a variable number of arguments, it is called with an argument list as defined by <stdarg.h>. *verrx()* does not return, but exits with the value of *eval*.

**Return Value**

None.

**Errors**

None.

**vfscanf**

**Name**

*vfscanf* — convert formatted input

**Description**

The *scanf()* family of functions shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except as noted below.

**Differences**

The %s, %S and %[ conversion specifiers shall accept an option length modifier a, which shall cause a memory buffer to be allocated to hold the string converted. In such a case, the argument corresponding to the conversion specifier should be a reference to a pointer value that will receive a pointer to the allocated buffer. If there is insufficient memory to allocate a buffer, the function may set *errno* to ENOMEM and a conversion error results.

**Note:** This directly conflicts with the ISO C (1999) usage of %a as a conversion specifier for hexadecimal float values. While this conversion specifier should be supported, a format specifier such as "%aseconds" will have a different meaning on an LSB conforming system.
vfscanf

Name
vfscanf — convert formatted input

Description
The scanf() family of functions shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except as noted below.

Differences
The %s, %S and %t conversion specifiers shall accept an option length modifier a, which shall cause a memory buffer to be allocated to hold the string converted. In such a case, the argument corresponding to the conversion specifier should be a reference to a pointer value that will receive a pointer to the allocated buffer. If there is insufficient memory to allocate a buffer, the function may set errno to ENOMEM and a conversion error results.

Note: This directly conflicts with the ISO C (1999) usage of %a as a conversion specifier for hexadecimal float values. While this conversion specifier should be supported, a format specifier such as "%aseconds" will have a different meaning on an LSB conforming system.

vscanf

Name
vscanf — convert formatted input

Description
The scanf() family of functions shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except as noted below.

Differences
The %s, %S and %t conversion specifiers shall accept an option length modifier a, which shall cause a memory buffer to be allocated to hold the string converted. In such a case, the argument corresponding to the conversion specifier should be a reference to a pointer value that will receive a pointer to the allocated buffer. If there is insufficient memory to allocate a buffer, the function may set errno to ENOMEM and a conversion error results.

Note: This directly conflicts with the ISO C (1999) usage of %a as a conversion specifier for hexadecimal float values. While this conversion specifier should be supported, a format specifier such as "%aseconds" will have a different meaning on an LSB conforming system.
vsscanf

Name
vsscanf — convert formatted input

Description
The scanf() family of functions shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except as noted below.

Differences
The %s, %S and %[ conversion specifiers shall accept an option length modifier a, which shall cause a memory buffer to be allocated to hold the string converted. In such a case, the argument corresponding to the conversion specifier should be a reference to a pointer value that will receive a pointer to the allocated buffer. If there is insufficient memory to allocate a buffer, the function may set errno to ENOMEM and a conversion error results.

Note: This directly conflicts with the ISO C (1999) usage of %a as a conversion specifier for hexadecimal float values. While this conversion specifier should be supported, a format specifier such as "%aseconds" will have a different meaning on an LSB conforming system.

vswscanf

Name
vswscanf — convert formatted input

Description
The scanf() family of functions shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except as noted below.

Differences
The %s, %S and %[ conversion specifiers shall accept an option length modifier a, which shall cause a memory buffer to be allocated to hold the string converted. In such a case, the argument corresponding to the conversion specifier should be a reference to a pointer value that will receive a pointer to the allocated buffer. If there is insufficient memory to allocate a buffer, the function may set errno to ENOMEM and a conversion error results.

Note: This directly conflicts with the ISO C (1999) usage of %a as a conversion specifier for hexadecimal float values. While this conversion specifier should be supported, a format specifier such as "%aseconds" will have a different meaning on an LSB conforming system.

vsyslog

Name
vsyslog — log to system log

Synopsis
#include <stdarg.h>
#include <syslog.h>
void vsyslog(int priority, char * message, va_list arglist);

Description
The vsyslog() function is identical to syslog() as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except that arglist (as defined by stdarg.h) replaces the variable number of arguments.

vwscanf

Name
vwscanf — convert formatted input

Description
The scanf() family of functions shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except as noted below.

Differences
The %s, %S and %[ conversion specifiers shall accept an option length modifier a, which shall cause a memory buffer to be allocated to hold the string converted. In such a case, the argument corresponding to the conversion specifier should be a reference to a pointer value that will receive a pointer to the allocated buffer. If there is insufficient memory to allocate a buffer, the function may set errno to ENOMEM and a conversion error results.

Note: This directly conflicts with the ISO C (1999) usage of %a as a conversion specifier for hexadecimal float values. While this conversion specifier should be supported, a format specifier such as "%aseconds" will have a different meaning on an LSB conforming system.

wait4

Name
wait4 — wait for process termination, BSD style

Synopsis
#include <sys/types.h>
#include <sys/resource.h>
#include <sys/wait.h>

pid_t wait4(pid_t pid, int *status, int options, struct rusage *rusage);

**Description**

`wait4()` suspends execution of the current process until a child (as specified by `pid`) has exited, or until a signal is delivered whose action is to terminate the current process or to call a signal handling function. If a child (as requested by `pid`) has already exited by the time of the call (a so-called “zombie” process), the function returns immediately. Any system resources used by the child are freed.

The value of `pid` can be one of:

- `< -1`
  - wait for any child process whose process group ID is equal to the absolute value of `pid`.

- `-1`
  - wait for any child process; this is equivalent to calling `wait3()`.

- `0`
  - wait for any child process whose process group ID is equal to that of the calling process.

- `> 0`
  - wait for the child whose process ID is equal to the value of `pid`.

The value of `options` is a bitwise or of zero or more of the following constants:

- **WNOHANG**
  - return immediately if no child is there to be waited for.

- **WUNTRACED**
  - return for children that are stopped, and whose status has not been reported.

If `status` is not NULL, `wait4()` stores status information in the location `status`. This status can be evaluated with the following macros:

- **WIFEXITED(status)**
  - is nonzero if the child exited normally.

- **WEXITSTATUS(status)**
  - evaluates to the least significant eight bits of the return code of the child that terminated, which may have been set as the argument to a call to `exit()` or as the argument for a return statement in the main program. This macro can only be evaluated if `WIFEXITED()` returned nonzero.

- **WIFSIGNALED(status)**
  - returns true if the child process exited because of a signal that was not caught.

- **WTERMSIG(status)**
  - returns the number of the signal that caused the child process to terminate. This macro can only be evaluated if `WIFSIGNALED()` returned nonzero.
WIFSTOPPED(status)
returns true if the child process that caused the return is currently stopped; this is only possible if the call was done using WUNTRACED().

WSTOPSIG(status)
returns the number of the signal that caused the child to stop. This macro can only be evaluated if WIFSTOPPED() returned nonzero.

If rusage is not NULL, the struct rusage (as defined in sys/resource.h) that it points to will be filled with accounting information. See getrusage() for details.

Return Value
On success, the process ID of the child that exited is returned. On error, -1 is returned (in particular, when no unwaited-for child processes of the specified kind exist), or 0 if WNOHANG() was used and no child was available yet. In the latter two cases, the global variable errno is set appropriately.

Errors
ECHILD
No unwaited-for child process as specified does exist.

ERESTARTSYS
A WNOHANG() was not set and an unblocked signal or a SIGCHLD was caught. This error is returned by the system call. The library interface is not allowed to return ERESTARTSYS, but will return EINTR.

warn
Name
warn — formatted error messages

Synopsis
#include <err.h>
void warn (const char *fmt , ...);

Description
The warn() function shall display a formatted error message on the standard error stream. The output shall consist of the last component of the program name, a colon character, and a space character. If fmt is non-NULL, it shall be used as a format string for the printf() family of functions, and the formatted message, a colon character, and a space are written to stderr. Finally, the error message string affiliated with the current value of the global variable errno shall be written to stderr, followed by a newline character.

Return Value
None.

Errors
None.
**warnx**

**Name**

`warnx` — formatted error messages

**Synopsis**

```c
#include <err.h>
void warnx (const char * fmt, ...);
```

**Description**

The `warnx()` function shall display a formatted error message on the standard error stream. The last component of the program name, a colon character, and a space shall be output. If `fmt` is non-NULL, it shall be used as the format string for the `printf()` family of functions, and the formatted error message, a colon character, and a space shall be output. The output shall be followed by a newline character.

**Return Value**

None.

**Errors**

None.

**wcstoq**

**Name**

`wcstoq` — convert wide string to long long int representation

**Synopsis**

```c
#include <wchar.h>
long long int wcstoq(const wchar_t * restrict nptr, wchar_t ** restrict endptr, int base);
```

**Description**

The `wcstoq()` function shall convert the initial portion of the wide string `nptr` to long long int representation. It is identical to `wcstoll()`.

**Return Value**

Refer to `wcstoll()`.

**Errors**

Refer to `wcstoll()`.
wcstouq

Name
wcstouq — convert wide string to unsigned long long int representation

Synopsis

#include <wchar.h>
unsigned long long wcstouq(const wchar_t * restrict nptr, wchar_t ** restrict endptr, int base);

Description
The wcstouq() function shall convert the initial portion of the wide string nptr to unsigned long long int representation. It is identical to wcstoull().

Return Value
Refer to wcstoull().

Errors
Refer to wcstoull().

wscanf

Name
wscanf — convert formatted input

Description
The scanf() family of functions shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except as noted below.

Differences
The %s, %S and %[ conversion specifiers shall accept an option length modifier a, which shall cause a memory buffer to be allocated to hold the string converted. In such a case, the argument corresponding to the conversion specifier should be a reference to a pointer value that will receive a pointer to the allocated buffer. If there is insufficient memory to allocate a buffer, the function may set errno to ENOMEM and a conversion error results.

Note: This directly conflicts with the ISO C (1999) usage of %a as a conversion specifier for hexadecimal float values. While this conversion specifier should be supported, a format specifier such as “%asseconds” will have a different meaning on an LSB conforming system.
xdr_u_int

Name

xdr_u_int — library routines for external data representation

Synopsis

int xdr_u_int(XDR * xdrs, unsigned int * up);

Description

xdr_u_int() is a filter primitive that translates between C unsigned integers and their external representations.

Return Value

On success, 1 is returned. On error, 0 is returned.

xdrstdio_create

Name

xdrstdio_create — library routines for external data representation

Synopsis

#include <rpc/xdr.h>
void xdrstdio_create(XDR * xdrs, FILE * file, enum xdr_op op);

Description

The xdrstdio_create() function shall initialize the XDR stream object referred to by xdrs. The XDR stream data shall be written to, or read from, the standard I/O stream associated with file. If the operation op is XDR_ENCODE, encoded data shall be written to file. If op is XDR_FREE, the XDR stream object may be used to deallocate storage allocated by a previous XDR_DECODE.

The associated destroy function shall flush the file I/O stream, but not close it.

Return Value

None.

14.6 Interfaces for libm

Table 14-38 defines the library name and shared object name for the libm library

<table>
<thead>
<tr>
<th>Table 14-38 libm Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Library: libm</td>
</tr>
<tr>
<td>SONAME: See architecture specific part.</td>
</tr>
</tbody>
</table>

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

[LSB] This Specification
14.6.1 Math

14.6.1.1 Interfaces for Math

An LSB conforming implementation shall provide the generic functions for Math specified in Table 14-39, with the full mandatory functionality as described in the referenced underlying specification.

### Table 14-39 libm - Math Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_finite [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>_finitef [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>_finitel [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>__fpclassify</td>
<td></td>
<td></td>
</tr>
<tr>
<td>__fpclassifyf</td>
<td></td>
<td></td>
</tr>
<tr>
<td>__signbit [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>__signbitf [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>acos [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>acosh [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>acoshf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>acoshl [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>acosl [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>asin [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>asinf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>asinh [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>asinhf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>asinhl [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>asinl [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>atan [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>atan2 [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>atan2f [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>atan2l [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>atanh [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>atanhf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>atanhf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cabs [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cabsf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cabsl [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cacos [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cacosf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cacosh [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cacoshf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cacoshl [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cacosl [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>carg [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>carg [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>carg [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>casin [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>casinf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>casinh [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>casinhf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>casinhl [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>catanh [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>catanhf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>catanh [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cos [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cos [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cexp [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cexpf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cexpl [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clog [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clog10 [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clog10f [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clog10l [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>conjugate [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>conj [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>conj [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cimag [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cimag [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>creal [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>crealf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cproj [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cproj [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cproj1 [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cproj1 [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>csqrt [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>csqrt [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>csqrt [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>csqrtf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>csqrtf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>csqrtf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>csqrtl [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>csqrt [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctan [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctan [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctanh [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctanh [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctanh [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drem [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dremf [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dremf [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drem [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dreml [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dreml [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>erf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>erfc [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>erfc [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>erfc [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>exp [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>exp10 [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>exp10f [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>exp10f [LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>exp2 [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>exp2f [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>exp2f [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>expm1 [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>expm1 [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>expm1 [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>expm1 [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fabs [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fabsf [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fabsl [SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function NAME (SUSv4)</td>
<td>Function NAME (LSB)</td>
<td>Function NAME (SUSv4)</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>i1 [SUSv4]</td>
<td>j1f [LSB]</td>
<td>j1f [LSB]</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
An LSB conforming implementation shall provide the generic deprecated functions for Math specified in Table 14-40, 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 14-40 libm - Math Deprecated Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>LSB</th>
<th>LSB</th>
<th>LSB</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>sqrtl</td>
<td>SUSv4</td>
<td>tan</td>
<td>SUSv4</td>
<td>tanh</td>
</tr>
<tr>
<td>tanhf</td>
<td>SUSv4</td>
<td>tanhl</td>
<td>SUSv4</td>
<td>tanl</td>
</tr>
<tr>
<td>tgammaf</td>
<td>SUSv4</td>
<td>tgammal</td>
<td>SUSv4</td>
<td>trunc</td>
</tr>
<tr>
<td>truncf</td>
<td>SUSv4</td>
<td>y0</td>
<td>SUSv4</td>
<td>y0f</td>
</tr>
<tr>
<td>y1</td>
<td>SUSv4</td>
<td>y1f</td>
<td>LSB</td>
<td>y1l</td>
</tr>
<tr>
<td>ynf</td>
<td>LSB</td>
<td>ynl</td>
<td>LSB</td>
<td></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Math specified in Table 14-41, with the full mandatory functionality as described in the referenced underlying specification.

### Table 14-41 libm - Math Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>LSB</th>
<th>LSB</th>
<th>LSB</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>signgam</td>
<td>SUSv4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 14.7 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.

#### 14.7.1 complex.h

```c
#define complex _Complex

extern double cabs(double complex);
extern float cabsf(float complex);
extern long double cabsl(long double complex);
extern double complex cacos(double complex);
extern float complex cacosf(float complex);
extern double complex cacosh(double complex);
extern float complex cacoshf(float complex);
extern long double complex cacoshl(long double complex);
```
extern long double complex cacosl(long double complex);
extern double carg(double complex);
extern float cargf(float complex);
extern long double complex cargl(long double complex);
extern double complex casin(double complex);
extern float complex casinf(float complex);
extern long double complex casinh(long double complex);
extern double complex casinhl(long double complex);
extern long double complex casinl(long double complex);
extern double complex catan(double complex);
extern float complex catanf(float complex);
extern long double complex catanh(long double complex);
extern double complex catanhl(long double complex);
extern long double complex catanl(long double complex);
extern double complex ccos(double complex);
extern float complex ccosf(float complex);
extern double complex ccosh(double complex);
extern float complex ccoshf(float complex);
extern long double complex ccoshl(long double complex);
extern double complex ccosl(long double complex);
extern double complex cexp(double complex);
extern float complex cexpf(float complex);
extern long double complex cexpl(long double complex);
extern double cimag(double complex);
extern float cimagf(float complex);
extern long double cimagl(long double complex);
extern double complex clog(double complex);
extern double complex clog10(double complex);
extern float complex clog10f(float complex);
extern long double complex clog10l(long double complex);
extern float complex clogf(float complex);
extern long double complex clogl(long double complex);
extern double complex conj(double complex);
extern float complex conjf(float complex);
extern long double complex conjl(long double complex);
extern double complex cpow(double complex, double complex);
extern float complex cpowf(float complex, float complex);
extern long double complex cpowl(long double complex, long double complex);
extern double complex cproj(double complex);
extern float complex cprojf(float complex);
extern long double complex cprojl(long double complex);
extern double creal(double complex);
extern float crealf(float complex);
extern long double creall(long double complex);
extern double complex csin(double complex);
extern float complex csinf(float complex);
extern double complex csinh(double complex);
extern float complex csinhf(float complex);
extern long double complex csinhl(long double complex);
extern double complex csinl(long double complex);
extern double complex csqrt(double complex);
extern float complex csqrtf(float complex);
extern long double complex csqrtl(long double complex);
extern double complex ctan(double complex);
extern float complex ctanf(float complex);
extern double complex ctanh(double complex);
extern float complex ctanhf(float complex);
extern long double complex ctanhl(long double complex);
extern long double complex ctanl(long double complex);

14.7.2 fenv.h
14 Base Libraries

14.7.3 math.h

#define DOMAIN 1
#define SING 2

#define FP_NAN 0
#define FP_INFINITY 1
#define FP_ZERO 2
#define FP_SUBNORMAL 3
#define FP_NORMAL 4

#define isnormal(x) (fpclassify (x) == FP_NORMAL) /* Return nonzero value if X is neither zero, subnormal, Inf, n */

#define HUGE_VAL 0x1.0p2047
#define HUGE_VALF 0x1.0p255f

#define NAN ((float)0x7fc00000UL)
#define M_1_PI 0.31830988618379067154
#define M_2_PI 0.63661977236758134308
#define M_LN2 0.69314718055994530942
#define M_SQRT1_2 0.70710678118654752440
#define M_PI_4 0.78539816339744834962
#define M_2_SQRTPI 1.12837916709551257390
#define M_SQRT2 1.41421356237309504880
#define M_LOG2E 1.4426950408889634074
#define M_PI_2 1.57079632679489661923
#define M_LN10 2.3025850929940456842
#define M_E 2.7182818284590452354
#define M_PI 3.14159265358979323846
#define INFINITY HUGE_VALF

#define MATH_ERRNO 1 /* errno set by math functions. */
#define MATH_ERREXCEPT 2 /* Exceptions raised by math functions. */

#define isunordered(u, v) __extension__((typeof( (u) ) __u = (u); typeof( (v) ) __v = (v); fpclassify (__u) != FP_NAN || fpclassify (__v) == FP_NAN; ))) /* Return nonzero value if arguments are unordered. */

#define islessgreater(x, y) __extension__((typeof( (x) ) __x = (x); typeof( (y) ) __y = (y); !isunordered ( __x, __y ) & ( __x < __y || __y < __x ); ))) /* Return nonzero value if either X is less than Y or Y is less */

#define isless(x, y)
(\_extension\_{{ \_typeof\_\_\_x = (x); \_typeof\_\_\_y = (y); !isunordered (\_x, \_y) && \_x < \_y; } }) \) /* Return nonzero value if \_x is less than \_y. */
#define islessequal(x, y) \ 
(\_extension\_{{ \_typeof\_\_\_x = (x); \_typeof\_\_\_y = (y); !isunordered (\_x, \_y) && \_x <= \_y; } })    /* Return nonzero value if \_x is less than or equal to \_y. */
#define isgreater(x,y)  \ 
(\_extension\_{{ \_typeof\_\_\_x = (x); \_typeof\_\_\_y = (y); !isunordered (\_x, \_y) && \_x > \_y; } })     /* Return nonzero value if \_x is greater than \_y. */
#define isgreaterequal(x,y)     \ 
(\_extension\_{{ \_typeof\_\_\_x = (x); \_typeof\_\_\_y = (y); !isunordered (\_x, \_y) && \_x >= \_y; } })    /* Return nonzero value if \_x is greater than or equal to \_y. */

extern int __finite(double);
extern int __finitef(float);
extern int __finitel(long double);
extern int __fpclassify(double);
extern int __fpclassifyf(float);
extern int __fpclassifyl(long double);
extern int __isinf(double);
extern int __isinff(float);
extern int __isinfl(long double);
extern int __isnan(double);
extern int __isnanf(float);
extern int __isnanl(long double);
extern int __signbit(double);
extern int __signbitf(float);
extern int __signbitl(long double);
extern double acos(double);
extern float acosf(float);
extern double acosh(double);
extern float acoshf(float);
extern long double acoshl(long double);
extern long double acosl(long double);
extern double asin(double);
extern float asinf(float);
extern double asinh(double);
extern float asinhf(float);
extern double asinhl(long double);
extern double asinl(long double);
extern double atan(double);
extern double atan2(double, double);
extern float atan2f(float, float);
extern long double atan2l(long double, long double);
extern float atanf(float);
extern double atanh(double);
extern float atanhf(float);
extern long double atanhf(long double);
extern long double atanhl(long double);
extern long double atanl(long double);
extern double cbrt(double);
extern float cbrtf(float);
extern long double cbrtl(long double);
extern double ceil(double);
extern float ceilf(float);
extern long double ceill(float);
#define cos(double)\
extern int __finite(double);
extern int __finitef(float);
extern int __finitel(long double);
extern int __fpclassify(double);
extern int __fpclassifyf(float);
extern int __fpclassifyl(long double);
extern int __isinf(double);
extern int __isinff(float);
extern int __isinfl(long double);
extern int __isnan(double);
extern int __isnanf(float);
extern int __isnanl(long double);
extern int __signbit(double);
extern int __signbitf(float);
extern int __signbitl(long double);
extern double acos(double);
extern float acosf(float);
extern double acosh(double);
extern float acoshf(float);
extern long double acoshl(long double);
extern long double cosl(long double);
extern double drem(double, double);
extern float dremf(float, float);
extern long double dreml(long double, long double);
extern double erf(double);
extern double erfc(double);
extern float erfcf(float);
extern long double erfcl(long double);
extern float erff(float);
extern long double erfl(long double);
extern double exp(double);
extern double exp10(double);
extern float exp10f(float);
extern long double exp10l(long double);
extern double exp2(double);
extern float exp2f(float);
extern float expf(float);
extern long double expl(long double);
extern double expm1(double);
extern float expm1f(float);
extern long double expm1l(long double);
extern double fabs(double);
extern float fabsf(float);
extern long double fabsl(long double);
extern double fdim(double, double);
extern float fdimf(float, float);
extern long double fdiml(long double, long double);
extern int finite(double);
extern int finitef(float);
extern int finitel(long double);
extern double floor(double);
extern float floorf(float);
extern long double floorl(long double);
extern double fma(double, double, double);
extern float fmaf(float, float, float);
extern long double fmal(long double, long double, long double);
extern double fmax(double, double);
extern float fmaxf(float, float);
extern long double fmaxl(long double, long double);
extern double fmin(double, double);
extern float fminf(float, float);
extern long double fminl(long double, long double);
extern double fmod(double, double);
extern float fmodf(float, float);
extern long double fmodl(long double, long double);
extern double frexp(double, int *);
extern float frexpf(float, int *);
extern long double frexpl(long double, int *);
extern double gamma(double);
extern float gammaf(float);
extern long double gammal(long double);
extern double hypot(double, double);
extern float hypotf(float, float);
extern long double hypotl(long double, long double);
extern int ilogb(double);
extern int ilogbf(float);
extern int ilogbl(long double);
extern double j0(double);
extern float j0f(float);
extern long double j0l(long double);
extern double j1(double);
extern float j1f(float);
extern long double j1l(long double);
extern double jn(int, double);
extern float jnf(int, float);
extern long double jnl(int, long double);
extern double ldexp(double, int);
extern float ldexpf(float, int);
extern long double ldexpl(long double, int);
extern double lgamma(double);
extern double lgamma_r(double, int *);
extern float lgammaf(float);
extern float lgammaf_r(float, int *);
extern long double lgammal(long double);
extern long double lgammal_r(long double, int *);
extern long int llrint(double);
extern long int llrintf(float);
extern long int llrintl(long double);
extern long int llround(double);
extern long int llroundf(float);
extern long int llroundl(long double);
extern double log10(double);
extern double log10f(float);
extern long double log10l(long double);
extern double log1p(double);
extern float log1pf(float);
extern long double log1pl(long double);
extern double log2(double);
extern float log2f(float);
extern long double log2l(long double);
extern double logb(double);
extern float logbf(float);
extern long double logbl(long double);
extern float logf(float);
extern long double log10l(long double);
extern float log10f(float);
extern double log1pl(long double);
extern float log1pf(float);
extern long double log1pl(long double);
extern float log1pf(float);
extern double log2l(long double);
extern float log2f(float);
extern long double log2l(long double);
extern double logbf(float);
extern float logbf(float);
extern double logf(float);
extern double logpl(long double);
extern float logpf(float);
extern double logpl(long double);
extern float logpf(float);
extern double log2pl(long double);
extern float log2pf(float);
extern double log2pl(long double);
extern double logbpl(long double);
extern float logbpf(float);
extern double logbpl(long double);
extern double logfpl(long double);
extern double remquo(double, double, int *);
extern float remquof(float, float, int *);
extern long double remquol(long double, long double, int *);
extern double fmod(double, double);
extern float fmodf(float, float);
extern long double fmodl(long double, long double);
extern double modf(double, double *);
extern float modff(float, float *);
extern long double modfl(long double, long double *);
extern double hypot(double, double);
extern float hypotf(float, float);
extern long double hypotl(long double, long double);
extern double isnan(double);
extern float isnanf(float);
extern long double isnanl(const char *);
extern double isinf(double);
extern float isinf(float);
extern long double isinfl(const char *);
extern double isfinite(double);
extern float isfinite(float);
extern long double isfinitepl(long double);
extern float isfinitef(float);
extern long double isfinitepl(long double);
extern double isunordered(double, double);
extern float isunorderedf(float, float);
extern long double isunorderedl(long double, long double);
extern double signbit(double);
extern float signbitf(float);
extern long double signbtl(const char *);
extern double copysign(double, double);
extern float copysignf(float, float);
extern long double copysignl(const char *);
extern double frexpl(long double *);
extern float frexpf(float *);
extern long double frexpll(long double *, long double *);
extern double ldexp(double, int);
extern float ldexpf(float, int);
extern long double ldexpll(long double *, long double *);
extern double modf(double, double *);
extern float modff(float, float *);
extern long double modfl,double *, double *);
extern double remquo(double, double, int *);
extern float remquof(float, float, int *);
extern long double remquol(long double, long double, int *);
extern double rint(double);
extern float rintf(float);
extern long double rintl(long double);
extern double round(double);
extern float roundf(float);
extern long double roundl(long double);
extern double remainder(double, double);
extern float remainderf(float, float);
extern long double remainderl(long double, long double);
extern double remquo(double, double, int *);
extern float remquof(float, float, int *);
extern long double remquol(long double, long double, int *);
extern doublerint(double);
extern floatrintf(float);
extern long doublerintl(long double);
extern doubleround(double);
extern doubleroundf(float);
extern long double roundl(long double);
extern double scalb(double, double);
extern float scalbf(float, double);
extern long double scalbl(long double, double, long double);
extern double scalbln(double, long int);
extern float scalblnf(float, long int);
extern long double scalblnl(long double, long int);
extern double scalbn(double, int);
extern float scalbnf(float, int);
extern long double scalbnl(long double, int);
extern int signgam;
extern double significand(double);
extern float significandf(float);
extern long double significandl(long double);
extern double sin(double);
extern void sincos(double, double *, double *);
extern void sincosf(float, float *, float *);
extern void sincosl(long double, double *, double *);
extern float sinf(float);
extern double sinh(double);
extern float sinhf(float);
extern long double sinhl(long double);
extern long double sinl(long double);
extern double sqrt(double);
extern float sqrtf(float);
extern long double sqrtl(long double);
extern double tan(double);
extern float tanf(float);
extern double tanh(double);
extern float tanhf(float);
extern long double tanhl(long double);
extern long double tanl(long double);
extern double tgamma(double);
extern float tgammaf(float);
extern long double tgammal(long double);
extern double trunc(double);
extern float truncf(float);
extern long double truncl(long double);
extern double y0(double);
extern float y0f(float);
extern long double y0l(long double);
extern double y1(double);
extern float y1f(float);
extern long double y1l(long double);
extern double yn(int, double);
extern float ynf(int, float);
extern long double ynl(int, long double);

### 14.8 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 14.6](#) shall behave as described in the referenced base document.
**__finite**

**Name**

__finite — test for infinity

**Synopsis**

```c
#include <math.h>
int __finite(double arg);
```

**Description**

__finite() has the same specification as isfinite() in POSIX_1003.1-2008 (ISO/IEC 9945-2009), except that the argument type for __finite() is known to be double.

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

**__finitef**

**Name**

__finitef — test for infinity

**Synopsis**

```c
#include <math.h>
int __finitef(float arg);
```

**Description**

__finitef() has the same specification as isfinite() in POSIX_1003.1-2008 (ISO/IEC 9945-2009) except that the argument type for __finitef() is known to be float.

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

**__finitel**

**Name**

__finitel — test for infinity

**Synopsis**

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

**Description**

__finitel() has the same specification as isfinite() in the POSIX_1003.1-2008 (ISO/IEC 9945-2009), except that the argument type for __finitel() is known to be long double.

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

Name
__fpclassify — Classify real floating type

Synopsis
int __fpclassify(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 double.
__fpclassify() is not in the source standard; it is only in the binary standard.

__fpclassifyf

Name
__fpclassifyf — Classify real floating type

Synopsis
int __fpclassifyf(float arg);

Description
__fpclassifyf() has the same specification as fpclassify() in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except that the argument type for __fpclassifyf() is known to be float.
__fpclassifyf() is not in the source standard; it is only in the binary standard.

__signbit

Name
__signbit — test sign of floating point value

Synopsis
#include <math.h>
int __signbit(double arg);

Description
__signbit() has the same specification as signbit() in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except that the argument type for __signbit() is known to be double.
__signbit() is not in the source standard; it is only in the binary standard.
__signbitf

Name
__signbitf — test sign of floating point value

Synopsis
#include <math.h>
int __signbitf(float arg);

Description
__signbitf() has the same specification as signbit() in POSIX 1003.1-2008
(ISO/IEC 9945-2009), except that the argument type for __signbitf() is known to be
float.
__signbitf() is not in the source standard; it is only in the binary standard.

clog10

Name
clog10 — Logarithm of a Complex Number

Synopsis
#include <complex.h>
double complex clog10(double complex z);

Description
The clog10() function shall compute the base 10 logarithm of the complex number z.

Return Value
The clog10() function shall return the base 10 logarithm.

clog10f

Name
clog10f — Logarithm of a Complex Number

Synopsis
#include <complex.h>
float complex clog10f(float complex z);

Description
The clog10f() function shall compute the base 10 logarithm of the complex number z.

Return Value
The clog10f() function shall return the base 10 logarithm.
clog10l

Name
clog10l — Logarithm of a Complex Number

Synopsis
#include <complex.h>
long double complex clog10l(long double complex z);

Description
The clog10l() function shall compute the base 10 logarithm of the complex number z.

Return Value
The clog10l() function shall return the base 10 logarithm.

drem

Name
drem — Floating Point Remainder (DEPRECATED)

Synopsis
#include <math.h>
double drem(double x, double y);

Description
The drem() function shall return the floating point remainder, x REM y as required by IEC 60559/IEEE 754 Floating Point in the same way as remainder().

Note: This function is included only for backwards compatibility; applications should use remainder() instead.

Returns
See remainder().

See Also
remainder(), dremf(), dreml()
dremf

Name
dremf — Floating Point Remainder (DEPRECATED)

Synopsis
#include <math.h>
double dremf(double x, double y);

Description
The dremf() function shall return the floating point remainder, \( x \text{ REM } y \) as required by IEC 60559/IEEE 754 Floating Point in the same way as remainderf().

Note: This function is included only for backwards compatibility; applications should use remainderf() instead.

Returns
See remainderf().

See Also
remainderf(), drem(), dreml()

dreml

Name
dreml — Floating Point Remainder (DEPRECATED)

Synopsis
#include <math.h>
double dreml(double x, double y);

Description
The dreml() function shall return the floating point remainder, \( x \text{ REM } y \) as required by IEC 60559/IEEE 754 Floating Point in the same way as remainderl().

Note: This function is included only for backwards compatibility; applications should use remainderl() instead.

Returns
See remainderl().

See Also
remainderl(), drem(), dremf()
exp10

Name
exp10 — Base-10 power function

Synopsis
#include <math.h>
double exp10(double x);

Description
The exp10() function shall return 10^x.

Note: This function is identical to pow10().

Returns
Upon successful completion, exp10() shall return 10 raised to the power of x.
If the correct value would cause overflow, a range error shall occur and exp10() shall return ±HUGE_VAL, with the same sign as the correct value of the function.

See Also
pow10(), exp10f(), exp10l()

exp10f

Name
exp10f — Base-10 power function

Synopsis
#include <math.h>
float exp10f(float x);

Description
The exp10f() function shall return 10^x.

Note: This function is identical to pow10f().

Returns
Upon successful completion, exp10f() shall return 10 raised to the power of x.
If the correct value would cause overflow, a range error shall occur and exp10f() shall return ±HUGE_VALF, with the same sign as the correct value of the function.

See Also
pow10f(), exp10(), exp10l()
exp10l

Name
exp10l — Base-10 power function

Synopsis
#include <math.h>
long double exp10l(long double x);

Description
The exp10l() function shall return 10^x.

Note: This function is identical to pow10l().

Returns
Upon successful completion, exp10l() shall return 10 raised to the power of x.
If the correct value would cause overflow, a range error shall occur and exp10l() shall return ±HUGE_VALL, with the same sign as the correct value of the function.

See Also
pow10l(), exp10(), exp10f()

fedisableexcept

Name
fedisableexcept — disable floating point exceptions

Synopsis
#include <fenv.h>
int fedisableexcept(int excepts);

Description
The fedisableexcept() function disables traps for each of the exceptions represented by the mask excepts.

Return Value
The fedisableexcept() function returns the previous set of enabled exceptions on success. On error, -1 is returned.

Errors
No errors are defined, but the function will fail if not supported on the architecture.
feenableexcept

Name
feenableexcept — enable floating point exceptions

Synopsis
#include <fenv.h>
int feenableexcept(int excepts);

Description
The feenableexcept() function enables traps for each of the exceptions represented by the mask excepts.

Return Value
The feenableexcept() function returns the previous set of enabled exceptions on success. On error, -1 is returned.

Errors
No errors are defined, but the function will fail if not supported on the architecture.

fegetexcept

Name
fegetexcept — query floating point exception handling state

Synopsis
#include <fenv.h>
int fegetexcept(void);

Description
The fegetexcept() function returns the set of all currently enabled exceptions.

Return Value
The fegetexcept() function returns the set of all currently enabled exceptions.

Errors
No errors are defined, but the function will fail if not supported on the architecture.

finite

Name
finite — test for infinity (DEPRECATED)

Synopsis
#define _SVID_SOURCE
#include <math.h>

int finite(double arg);

**Description**

The `finite()` function shall test whether its argument is neither INFINITY nor not a number (NaN).

**Returns**

On success, `finite()` shall return 1. Otherwise the function shall return 0.

*Note:* The ISO C (1999) standard defines the function `isfinite()`, which is more general purpose. The `finite()` function is deprecated, and applications should use `isfinite()` instead. A future revision of this standard may remove this function.

**See Also**

`isfinite()`, `finitef()`, `finitel()`

### finitef

**Name**

`finitef` — test for infinity (DEPRECATED)

**Synopsis**

```c
#define _SVID_SOURCE
#include <math.h>
int finitef(float arg);
```

**Description**

The `finitef()` function shall test whether its argument is neither INFINITY nor not a number (NaN).

**Returns**

On success, `finitef()` shall return 1. Otherwise the function shall return 0.

*Note:* The ISO C (1999) standard defines the function `isfinite()`, which is more general purpose. The `finitef()` function is deprecated, and applications should use `isfinite()` instead. A future revision of this standard may remove this function.

**See Also**

`isfinite()`, `finite()`, `finitel()`

### finitel

**Name**

`finitel` — test for infinity (DEPRECATED)

**Synopsis**

```c
#define _SVID_SOURCE
```

© 2015 Linux Foundation
#include <math.h>
int finitel(long double arg);

**Description**
The `finitel()` function shall test whether its argument is neither INFINITY nor not a number (NaN).

**Returns**
On success, `finitel()` shall return 1. Otherwise the function shall return 0.

*Note:* The [ISO C (1999)] standard defines the function `isfinite()`, which is more general purpose. The `finitel()` function is deprecated, and applications should use `isfinite()` instead. A future revision of this standard may remove this function.

**See Also**
`isfinite()`, `finite()`, `finitef()`

gamma

**Name**
gamma — log gamma function (DEPRECATED)

**Synopsis**
```c
#include <math.h>
double gammaf(double x);
```

**Description**
The `gamma()` function is identical to `lgamma()` in [POSIX 1003.1-2008 (ISO/IEC 9945-2009)].

*Note:* The name `gamma()` for this function is deprecated and should not be used.

**Returns**
See `lgamma()`.

**See Also**
`lgamma()`, `lgammaf()`, `lgammal()`, `gammaf()`, `gammal()`
gammaf

**Name**
gammaf — log gamma function (DEPRECATED)

**Synopsis**
#include <math.h>
float gammaf(float x);

**Description**
The gammaf() function is identical to lgammaf() in POSIX 1003.1-2008 (ISO/IEC 9945-2009).

*Note:* The name gammaf() for this function is deprecated and should not be used.

**Returns**
See lgammaf().

**See Also**
lgamma(), lgammaf(), lgammal(), gamma(), gammal()

gammal

**Name**
gammal — log gamma function (DEPRECATED)

**Synopsis**
#include <math.h>
long double gammal(long double x);

**Description**
The gammal() function is identical to lgammal() in POSIX 1003.1-2008 (ISO/IEC 9945-2009).

*Note:* The name gammal() for this function is deprecated and should not be used.

**Returns**
See lgammal().

**See Also**
lgamma(), lgammaf(), lgammal(), gamma(), gammal()
j0f

Name
j0f — Bessel functions

Synopsis
#include <math.h>
float j0f(float x);

Description
The j0f() function is identical to j0(), except that the argument x and the return value
is a float.

Returns
See j0().

See Also
j0(). j0l(). j1(). j1f(). j1l(). jn(). jnf(). jnl(). y0(). y0f(). y0l(). y1().
y1f(). y1l(). yn(). ynf(). ynl()

j0l

Name
j0l — Bessel functions

Synopsis
#include <math.h>
long double j0l(long double x);

Description
The j0l() function is identical to j0(), except that the argument x and the return value
is a long double.

Returns
See j0().

See Also
j0(). j0f(). j1(). j1f(). j1l(). jn(). jnf(). jnl(). y0(). y0f(). y0l(). y1().
y1f(). y1l(). yn(). ynf(). ynl()
**j1f**

**Name**

j1f — Bessel functions

**Synopsis**

```c
#include <math.h>
float j1f(float x);
```

**Description**

The j1f() function is identical to j1(), except that the argument x and the return value is a float.

**Returns**

See j1().

**See Also**

j0(), j0f(), j0l(), j1(), j1f(), j1l(), jn(), jnf(), jnl(), y0(), y0f(), y0l(), y1(), y1f(), y1l(), yn(), ynf(), ynl()
**jnf**

**Name**

jnf — Bessel functions

**Synopsis**

```c
#include <math.h>
float jnf(float x);
```

**Description**

The jnf() function is identical to jn(), except that the argument x and the return value is a float.

**Returns**

See jn().

**See Also**

j0(), j0f(), j0l(), j1(), j1f(), j1l(), jn(), jnl(), jn0(), jn0f(), jn0l(), jn1(), jn1f(), jn1l(), y0(), y0f(), y0l(), y1(), y1f(), y1l(), yn(), ynf(), ynl()

**jnl**

**Name**

jnl — Bessel functions

**Synopsis**

```c
#include <math.h>
long double jnl(long double x);
```

**Description**

The jnl() function is identical to jn(), except that the argument x and the return value is a long double.

**Returns**

See jn().

**See Also**

j0(), j0f(), j0l(), j1(), j1f(), j1l(), jn(), jnf(), y0(), y0f(), y0l(), y1(), y1f(), y1l(), yn(), ynf(), ynl()
**lgamma_r**

**Name**

*lgamma_r* — log gamma functions

**Synopsis**

```
#include <math.h>
double lgamma_r(double x, int * signp);
```

**Description**

The *lgamma_r()* function shall compute the natural logarithm of the absolute value of the Gamma function, as *lgamma()*. However, instead of setting the external integer *signgam* to the sign of the Gamma function, *lgamma_r()* shall set the integer referenced by *signp* to the sign.

**Returns**

See *lgamma()* and *signgam*.

**See Also**

*lgamma(), lgammaf_r(), lgamml_r(), signgam*

---

**lgammaf_r**

**Name**

*lgammaf_r* — log gamma functions

**Synopsis**

```
#include <math.h>
float lgammaf_r(float x, int * signp);
```

**Description**

The *lgammaf_r()* function shall compute the natural logarithm of the absolute value of the Gamma function, as *lgammaf()*. However, instead of setting the external integer *signgam* to the sign of the Gamma function, *lgammaf_r()* shall set the integer referenced by *signp* to the sign.

**Returns**

See *lgammaf()* and *signgam*.

**See Also**

*lgamma(), lgamma_r(), lgamml_r(), signgam*
**lgammal_r**

**Name**

*lgammal_r* — log gamma functions

**Synopsis**

```c
#include <math.h>
double lgammal_r(double x, int *signp);
```

**Description**

The *lgammal_r()* function shall compute the natural logarithm of the absolute value of the Gamma function, as *lgamma()*. However, instead of setting the external integer *signgam* to the sign of the Gamma function, *lgammal_r()* shall set the integer referenced by *signp* to the sign.

**Returns**

See *lgamma()* and *signgam*.

**See Also**

*lgamma()*.*lgammal_r()*.*lgammaf_r()*.*signgam*

---

**matherr**

**Name**

*matherr* — math library exception handling

**Synopsis**

```c
#include <math.h>
int matherr(struct exception *__exc);
```

**Description**

The System V Interface Definition (SVID) Issue 3 specifies that various math functions should invoke a function called *matherr()* if a math exception is detected. This function is called before the math function returns; after *matherr()* returns, the system then returns to the math function, which in turn returns to the caller.

*matherr()* is obsolete; indeed it was withdrawn in the System V Interface Definition (SVID) Issue 4, and is required only by this specification for historical compatibility, and will be removed in a future version. The floating point environment function group including *fesetenv()* should be used instead.

*matherr()* is not in the source standard; it is only in the binary standard.

**See Also**

*fesetenv()*.*fegetenv()*.*feupdateenv()*.
**pow10**

**Name**

`pow10` — Base-10 power function

**Synopsis**

```c
#include <math.h>
double pow10(double x);
```

**Description**

The `pow10()` function shall return $10^x$.

*Note:* This function is identical to `exp10()`.

**Returns**

Upon successful completion, `pow10()` shall return $10$ raised to the power of `$x$.

If the correct value would cause overflow, a range error shall occur and `pow10()` shall return ±HUGE_VAL, with the same sign as the correct value of the function.

**See Also**

`exp10()`, `pow10f()`, `pow10l()`

**pow10f**

**Name**

`pow10f` — Base-10 power function

**Synopsis**

```c
#include <math.h>
float pow10f(float x);
```

**Description**

The `pow10f()` function shall return $10^x$.

*Note:* This function is identical to `exp10f()`.

**Returns**

Upon successful completion, `pow10f()` shall return $10$ raised to the power of `$x$.

If the correct value would cause overflow, a range error shall occur and `pow10f()` shall return ±HUGE_VALF, with the same sign as the correct value of the function.

**See Also**

`exp10f()`, `pow10()`, `pow10l()`
**pow10l**

**Name**

`pow10l` — Base-10 power function

**Synopsis**

```c
#include <math.h>
long double pow10l(long double x);
```

**Description**

The `pow10l()` function shall return $10^x$.

**Note:** This function is identical to `exp10l()`.

**Returns**

Upon successful completion, `pow10l()` shall return $10$ raised to the power of $x$.

If the correct value would cause overflow, a range error shall occur and `pow10l()` shall return ±HUGE_VALL, with the same sign as the correct value of the function.

**See Also**

`exp10l()`, `pow10()`, `pow10f()`

**scalbf**

**Name**

`sclafb` — load exponent of radix-independent floating point number

**Synopsis**

```c
#include <math.h>
float scalbf(float x, double exp);
```

**Description**

The `scalbf()` function is identical to `scalb()`, except that the argument $x$ and the return value is of type `float`.

**Returns**

See `scalb()`.
scalbl

Name
scalbl — load exponent of radix-independent floating point number

Synopsis
#include <math.h>
long double scalbl(long double x, double exp);

Description
The scalbl() function is identical to scalb(), except that the argument x and the return value is of type long double.

Returns
See scalb().

significand

Name
significand — floating point mantissa

Synopsis
#include <math.h>
double significand(double x);

Description
The significand() function shall return the mantissa of x, sig such that x ≡ sig × 2^n scaled such that 1 ≤ sig < 2.

Note: This function is intended for testing conformance to IEEE 754 Floating Point, and its use is not otherwise recommended.
This function is equivalent to scalb(x, (double)-ilogb(x)).

Returns
Upon successful completion, significand() shall return the mantissa of x in the range 1 ≤ sig < 2.
If x is 0, ±HUGE_VAL, or NaN, the result is undefined.

See Also
significandf(), significandl()
significandf

**Name**
significandf — floating point mantissa

**Synopsis**

```c
#include <math.h>
float significandf(float x);
```

**Description**

The significandf() function shall return the mantissa of \( x \), \( \text{sig} \) such that \( x \equiv \text{sig} \times 2^\text{n} \) scaled such that \( 1 \leq \text{sig} < 2 \).

**Note:** This function is intended for testing conformance to IEC 60559/IEEE 754 Floating Point, and its use is not otherwise recommended.

This function is equivalent to `scalb(x, (double)-ilogb(x))`.

**Returns**

Upon successful completion, significandf() shall return the mantissa of \( x \) in the range \( 1 \leq \text{sig} < 2 \).

If \( x \) is 0, ±HUGE_VALF, or NaN, the result is undefined.

**See Also**

significand(), significandl()
**significandl**

**Name**

`significandl` — floating point mantissa

**Synopsis**

```c
#include <math.h>
long double significandl(long double x);
```

**Description**

The `significandl()` function shall return the mantissa of `x`, `sig` such that `x ≡ sig × 2^n` scaled such that `1 ≤ sig < 2`.

**Note:** This function is intended for testing conformance to [IEC 60559/IEEE 754 Floating Point](https://www.iec.ch), and its use is not otherwise recommended.

This function is equivalent to `scalb(x, (double)-ilogb(x))`.

**Returns**

Upon successful completion, `significandl()` shall return the mantissa of `x` in the range `1 ≤ sig < 2`.

If `x` is 0, ±HUGE_VALL, or NaN, the result is undefined.

**See Also**

`significand()`, `significandf()`

**sincos**

**Name**

`sincos` — trigonometric functions

**Synopsis**

```c
#include <math.h>
#include <_GNU_SOURCE>
void sincos(double x, double *sin, double *cos);
```

**Description**

The `sincos()` function shall calculate both the sine and cosine of `x`. The sine shall be stored in the location referenced by `sin`, and the cosine in the location referenced by `cosine`.

**Returns**

None. See `sin()` and `cos()` for possible error conditions.

**See Also**

`cos()`, `sin()`, `sincosf()`, `sincosl()`
sincosf

Name
sincosf — trigonometric functions

Synopsis
#define _GNU_SOURCE
#include <math.h>
void sincosf(float x, float *sin, float *cos);

Description
The sincosf() function shall calculate both the sine and cosine of x. The sine shall be
stored in the location referenced by sin, and the cosine in the location referenced by
cosine.

Returns
None. See sin() and cos() for possible error conditions.

See Also
cos().sin().sincos().sincosl()

sincosl

Name
sincosl — trigonometric functions

Synopsis
#define _GNU_SOURCE
#include <math.h>
void sincosl(long double x, long double *sin, long double *cos);

Description
The sincosl() function shall calculate both the sine and cosine of x. The sine shall be
stored in the location referenced by sin, and the cosine in the location referenced by
cosine.

Returns
None. See sin() and cos() for possible error conditions.

See Also
cos().sin().sincos().sincosl()
**y0f**

**Name**

y0f — Bessel functions

**Synopsis**

```c
#include <math.h>
float y0f(float x);
```

**Description**

The y0f() function is identical to y0(), except that the argument x and the return value is a float.

**Returns**

See y0().

**See Also**

j0(), j0f(), j0l(), j1(), j1f(), j1l(), jn(), jnf(), jnl(), y0(), y0f(), y0l(). y1(), y1f(), y1l(), yn(), ynf(), ynl()
**y1f**

**Name**

*y1f* — Bessel functions

**Synopsis**

```c
#include <math.h>
float y1f(float x);
```

**Description**

The *y1f()* function is identical to *y1()*, except that the argument *x* and the return value is a float.

**Returns**

See *y1()*. 

**See Also**

*j0(), j0f(), j0l(), j1(), j1f(), j1l(), jn(), jnf(), jnl(), y0(), y0f(), y0l(), y1(), y1f(), y1l(), yn(), ynf(), ynl()*

**y1l**

**Name**

*y1l* — Bessel functions

**Synopsis**

```c
#include <math.h>
long double y1l(long double x);
```

**Description**

The *y1l()* function is identical to *y1()*, except that the argument *x* and the return value is a long double.

**Returns**

See *j0()*. 

**See Also**

*j0(), j0f(), j0l(), j1(), j1f(), j1l(), jn(), jnf(), jnl(), y0(), y0f(), y0l(), y1(), y1f(), y1l(), yn(), ynf(), ynl()*
ynf

Name

ynf — Bessel functions

Synopsis

#include <math.h>
float yn(float x);

Description

The yn() function is identical to y0(), except that the argument x and the return value is a float.

Returns

See y0().

See Also

j0(), j0f(), j0l(), j1(), j1f(), j1l(), jn(), jnf(), jnl(), y0(), y0f(), y0l(), y1(), y1f(), y1l(), yn(), ynl()

ynl

Name

ynl — Bessel functions

Synopsis

#include <math.h>
long double ynl(long double x);

Description

The ynl() function is identical to y0(), except that the argument x and the return value is a long double.

Returns

See y0().

See Also

j0(), j0f(), j0l(), j1(), j1f(), j1l(), jn(), jnf(), jnl(), y0(), y0f(), y0l(), y1(), y1f(), y1l(), yn(), ynl()

14.9 Interfaces for libpthread

Table 14-42 defines the library name and shared object name for the libpthread library

<table>
<thead>
<tr>
<th>Library:</th>
<th>libpthread</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libpthread.so.0</td>
</tr>
</tbody>
</table>

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

14.9.1.1 Interfaces for Realtime Threads

An LSB conforming implementation shall provide the generic functions for Realtime Threads specified in Table 14-43, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-43 libpthread - Realtime Threads Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>pthread_attr_getinheritsched</td>
<td>Obtains the inheritance policy of a thread</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_attr_getschedpolicy</td>
<td>Obtains the scheduling policy of a thread</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_attr_getprioceiling(GLIBC_2.4)</td>
<td>Obtains the priority ceiling of a thread</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_setschedparam(GLIBC_2.3.4)</td>
<td>Sets the scheduling parameters of a thread</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_attr_setinheritsched</td>
<td>Sets the inheritance policy of a thread</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_attr_setschedpolicy</td>
<td>Sets the scheduling policy of a thread</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_attr_setschedpolicy</td>
<td>Sets the scheduling policy of a thread</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_setschedparam(GLIBC_2.3.4)</td>
<td>Sets the scheduling parameters of a thread</td>
<td>[SUSv4]</td>
</tr>
</tbody>
</table>

14.9.2 Advanced Realtime Threads

14.9.2.1 Interfaces for Advanced Realtime Threads

An LSB conforming implementation shall provide the generic functions for Advanced Realtime Threads specified in Table 14-44, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-44 libpthread - Advanced Realtime Threads Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>pthread_barrier_destroy</td>
<td>Destroys a barrier</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_barrierattr_getpshared(GLIBC_2.3.3)</td>
<td>Gets the shared attribute of a barrier</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_barrierattr_init</td>
<td>Initializes a barrier</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_barrierattr_setpshared</td>
<td>Sets the shared attribute of a barrier</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_getcpuclockid</td>
<td>Gets the CPU clock id of a thread</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_spin_destroy</td>
<td>Destroys a spin lock</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_spin_init</td>
<td>Initializes a spin lock</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_spin_lock</td>
<td>Locks a spin lock</td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>pthread_spin_unlock</td>
<td>Unlocks a spin lock</td>
<td>[SUSv4]</td>
</tr>
</tbody>
</table>

14.9.3 Posix Threads

14.9.3.1 Interfaces for Posix Threads

An LSB conforming implementation shall provide the generic functions for Posix Threads specified in Table 14-45, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-45 libpthread - Posix Threads Function Interfaces
<table>
<thead>
<tr>
<th>Function</th>
<th>LSB</th>
<th>SUSv4</th>
<th>SUSv3</th>
</tr>
</thead>
<tbody>
<tr>
<td>pthread_attr_getargsize</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_getschedparam</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_getstack</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_getstackaddr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_getstacksize</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_getstacktstate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_init</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_setstacktstate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_setstackaddr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_setstacksize</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_setschedparam</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_setstack</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_setstackaddr</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_setstacksize</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_attr_setstacktstate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_cancel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_cond_broadcast</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_cond_destroy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_cond_getpshared</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_cond_gettinit</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_cond_timedwait</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutex_consistent</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutex_destroy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutex_init</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutex_lock</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutex_timedlock</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutex_trylock</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutex_unlock</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutexattr_destroy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutexattr_getpshared</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutexattr_getrobust</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutexattr_gettype</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutexattr_init</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutexattr_setpshared</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutexattr_setrobust</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_mutexattr_settype</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_once</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlock_destroy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlock_init</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlock_rdlock</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlock_timedrdlock</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlock_tryrdlock</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlock_uunlock</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlockattr_destroy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlockattr_getpshared</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlockattr_getrobust</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlockattr_gettype</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlockattr_init</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlockattr_setpshared</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlockattr_setrobust</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_rwlockattr_settype</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_self</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_setspecific</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pthread_setsid</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sem_destroy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sem_getvalue</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic deprecated functions for Posix Threads specified in Table 14-46, 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 14-46 libpthread - Posix Threads Deprecated Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sem_post</td>
<td></td>
</tr>
<tr>
<td>sem_timedwait</td>
<td></td>
</tr>
<tr>
<td>sem_trywait</td>
<td></td>
</tr>
<tr>
<td>sem_unlink</td>
<td></td>
</tr>
</tbody>
</table>

14.9.4 Thread aware versions of libc interfaces

14.9.4.1 Interfaces for Thread aware versions of libc interfaces

An LSB conforming implementation shall provide the generic functions for Thread aware versions of libc interfaces specified in Table 14-47, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-47 libpthread - Thread aware versions of libc interfaces Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lseek64</td>
<td></td>
</tr>
<tr>
<td>open64</td>
<td></td>
</tr>
<tr>
<td>pread</td>
<td></td>
</tr>
<tr>
<td>pread64</td>
<td></td>
</tr>
<tr>
<td>pwrite</td>
<td></td>
</tr>
<tr>
<td>pwrite64</td>
<td></td>
</tr>
</tbody>
</table>

14.9.5 GNU Extensions for libpthread

14.9.5.1 Interfaces for GNU Extensions for libpthread

An LSB conforming implementation shall provide the generic functions for GNU Extensions for libpthread specified in Table 14-48, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-48 libpthread - GNU Extensions for libpthread Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pthread_getattr_np</td>
<td></td>
</tr>
<tr>
<td>pthread_mutex_consistent_np</td>
<td></td>
</tr>
<tr>
<td>pthread_mutexattr_getrobust_np</td>
<td></td>
</tr>
<tr>
<td>pthread_mutexattr_setrobust_np</td>
<td></td>
</tr>
<tr>
<td>pthread_rwlockattr_getkind_np</td>
<td></td>
</tr>
<tr>
<td>pthread_rwlockattr_setkind_np</td>
<td></td>
</tr>
</tbody>
</table>

14.9.6 System Calls

14.9.6.1 Interfaces for System Calls

An LSB conforming implementation shall provide the generic functions for System Calls specified in Table 14-49, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-49 libpthread - System Calls Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>close</td>
<td></td>
</tr>
<tr>
<td>fcntl</td>
<td></td>
</tr>
<tr>
<td>fork</td>
<td></td>
</tr>
<tr>
<td>fsync</td>
<td></td>
</tr>
<tr>
<td>lseek</td>
<td></td>
</tr>
<tr>
<td>msync</td>
<td></td>
</tr>
<tr>
<td>nanosleep</td>
<td></td>
</tr>
<tr>
<td>open</td>
<td></td>
</tr>
</tbody>
</table>
14.9.7 Standard I/O

14.9.7.1 Interfaces for Standard I/O

An LSB conforming implementation shall provide the generic functions for Standard I/O specified in Table 14-50, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-50 libpthread - Standard I/O Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>flockfile</td>
<td>SUSv4</td>
</tr>
<tr>
<td>read</td>
<td>SUSv4</td>
</tr>
<tr>
<td>vfork</td>
<td>SUSv3</td>
</tr>
<tr>
<td>wait</td>
<td>SUSv4</td>
</tr>
<tr>
<td>waitpid</td>
<td>LSB</td>
</tr>
<tr>
<td>write</td>
<td>SUSv4</td>
</tr>
</tbody>
</table>

14.9.8 Signal Handling

14.9.8.1 Interfaces for Signal Handling

An LSB conforming implementation shall provide the generic functions for Signal Handling specified in Table 14-51, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-51 libpthread - Signal Handling Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>__libc_current_sig rtmax</td>
<td>LSB</td>
</tr>
<tr>
<td>__libc_current_sigtmin</td>
<td>LSB</td>
</tr>
<tr>
<td>raise</td>
<td>SUSv4</td>
</tr>
<tr>
<td>sigaction</td>
<td>SUSv4</td>
</tr>
<tr>
<td>siglongjmp</td>
<td>SUSv4</td>
</tr>
<tr>
<td>sigwait</td>
<td>SUSv4</td>
</tr>
</tbody>
</table>

14.9.9 Standard Library

14.9.9.1 Interfaces for Standard Library

An LSB conforming implementation shall provide the generic functions for Standard Library specified in Table 14-52, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-52 libpthread - Standard Library Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>__errno_location</td>
<td>LSB</td>
</tr>
<tr>
<td>ftrylockfile</td>
<td>SUSv4</td>
</tr>
<tr>
<td>funlockfile</td>
<td>SUSv4</td>
</tr>
<tr>
<td>longjmp</td>
<td>SUSv4</td>
</tr>
<tr>
<td>system</td>
<td>LSB</td>
</tr>
</tbody>
</table>

14.9.10 Socket Interface

14.9.10.1 Interfaces for Socket Interface

An LSB conforming implementation shall provide the generic functions for Socket Interface specified in Table 14-53, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-53 libpthread - Socket Interface Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>__h_errno_location</td>
<td>LSB</td>
</tr>
<tr>
<td>accept</td>
<td>SUSv4</td>
</tr>
<tr>
<td>connect</td>
<td>SUSv4</td>
</tr>
<tr>
<td>recv</td>
<td>SUSv4</td>
</tr>
<tr>
<td>recvfrom</td>
<td>SUSv4</td>
</tr>
<tr>
<td>recvmmsg</td>
<td>SUSv4</td>
</tr>
<tr>
<td>send</td>
<td>SUSv4</td>
</tr>
<tr>
<td>sendmsg</td>
<td>SUSv4</td>
</tr>
<tr>
<td>sendto</td>
<td>SUSv4</td>
</tr>
</tbody>
</table>
14.9.11 Terminal Interface Functions

14.9.11.1 Interfaces for Terminal Interface Functions

An LSB conforming implementation shall provide the generic functions for Terminal Interface Functions specified in Table 14-54, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-54 libpthread - Terminal Interface Functions Function Interfaces

| tcdrain [SUSv4] |

14.10 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.

14.10.1 lsb/pthread.h

typedef unsigned long int pthread_t;

14.10.2 pthread.h

#define PTHREAD_MUTEX_DEFAULT 0
#define PTHREAD_MUTEX_NORMAL 0
#define PTHREAD_SCOPE_SYSTEM 0
#define PTHREAD_MUTEX_RECURSIVE 1
#define PTHREAD_SCOPE_PROCESS 1
#define PTHREAD_MUTEX_ERRORCHECK 2
#define __SIZEOF_PTHREAD_BARRIERATTR_T 4
#define __SIZEOF_PTHREAD_CONDATTR_T 4
#define __SIZEOF_PTHREAD_MUTEXATTR_T 4
#define __SIZEOF_PTHREAD_COND_T 48
#define __SIZEOF_PTHREAD_RWLOCKATTR_T 8
#define pthread_cleanup_push(routine,arg) \ 
{struct _pthread_cleanup_buffer _buffer; \ _pthread_cleanup_push(&_buffer,(routine),(arg));
#define pthread_cleanup_pop(execute) \ 
(pthread_cleanup_pop(&buffer,(execute));)
#define PTHREAD_COND_INITIALIZER { { 0, 0, 0, 0, 0, (void *) 0, 0, 0 } }

struct _pthread_cleanup_buffer {
    void *__routine (void *);
    void *__arg;
    int __canceltype;
    struct _pthread_cleanup_buffer *__prev;
};
typedef unsigned int pthread_key_t;
typedef int pthread_once_t;
typedef volatile int pthread_spinlock_t;
typedef union {
    char __size[__SIZEOF_PTHREAD_BARRIERATTR_T];
    int __align;
} pthread_barrierattr_t;
enum {
    PTHREAD_PRIO_NONE,
    PTHREAD_PRIO_INHERIT,
    PTHREAD_PRIO_PROTECT
};
enum {
    PTHREAD_MUTEX_STALLED = 0,
    PTHREAD_MUTEX_STALLED_NP = 0,
    PTHREAD_MUTEX_ROBUST = 1,
    PTHREAD_MUTEX_ROBUST_NP = 1
};
enum {
    PTHREAD_RWLOCK_PREFER_READER_NP,
    PTHREAD_RWLOCK_PREFER_WRITER_NP,
    PTHREAD_RWLOCK_PREFER_WRITER_NONRECURSIVE_NP,
    PTHREAD_RWLOCK_DEFAULT_NP = PTHREAD_RWLOCK_PREFER_READER_NP
};
typedef union {
    struct __pthread_mutex_s __data;
    char __size[__SIZEOF_PTHREAD_MUTEX_T];
    long int __align;
} pthread_mutex_t;
typedef union {
    char __size[__SIZEOF_PTHREAD_MUTEXATTR_T];
    int __align;
} pthread_mutexattr_t;
typedef union {
    char __size[__SIZEOF_PTHREAD_ATTR_T];
    long int __align;
} pthread_attr_t;
typedef union {
    struct {
        int __lock;
        unsigned int __futex;
        unsigned long long int __total_seq;
        unsigned long long int __wakeup_seq;
        unsigned long long int __woken_seq;
        void *__mutex;
        unsigned int __nwaiters;
        unsigned int __broadcast_seq;
    } __data;
    char __size[__SIZEOF_PTHREAD_COND_T];
    long long int __align;
} pthread_cond_t;
typedef union {
    char __size[__SIZEOF_PTHREAD_CONDATTR_T];
    int __align;
} pthread_condattr_t;
typedef union {
    char __size[__SIZEOF_PTHREAD_RWLOCKATTR_T];
    long int __align;
} pthread_rwlockattr_t;
#define PTHREAD_CREATE_JOINABLE 0
#define PTHREAD_INHERIT_SCHED 0
#define PTHREAD_ONCE_INIT 0
#define PTHREAD_PROCESS_PRIVATE 0
#define PTHREAD_CREATE_DETACHED 1
#define PTHREAD_EXPLICIT_SCHED 1
#define PTHREAD_PROCESS_SHARED 1
#define PTHREAD_CANCELED ((void*)-1)
#define PTHREAD_CANCEL_DEFERRED 0
#define PTHREAD_CANCEL_ENABLE 0
#define PTHREAD_CANCEL_ASYNCHRONOUS 1
#define PTHREAD_CANCEL_DISABLE 1

extern int __register_atfork(void (*)(void), void (*)(void),
                           void (*)(void), void *);
extern void _pthread_cleanup_pop(struct _pthread_cleanup_buffer *), int);
extern void _pthread_cleanup_push(struct _pthread_cleanup_buffer *),
            void (*)(void *), void *);
extern int pthread_atfork(void (*__prepare) (void),
                          void (*__parent) (void), void (*__child) (void));
extern int pthread_attr_destroy(pthread_attr_t * __attr);
extern  int  pthread_attr_getdetachstate(const  pthread_attr_t  * __attr,
                                        int *__detachstate);
extern  int  pthread_attr_getguardsize(const  pthread_attr_t  * __attr,
                                        size_t * __guardsize);
extern  int  pthread_attr_getinheritsched(const  pthread_attr_t  * __attr,
                                        int * __inherit);
extern  int  pthread_attr_getschedparam(const  pthread_attr_t  * __attr,
                                        struct sched_param * __param);
extern int pthread_attr_getschedpolicy(const pthread_attr_t * __attr,
                                       int * __policy);
extern int pthread_attr_getscope(const pthread_attr_t * __attr,
                                      int * __scope);
extern int pthread_attr_getstack(const pthread_attr_t * __attr,
                                      void **__stackaddr, size_t * __stacksize);
extern int pthread_attr_getstackaddr(const pthread_attr_t * __attr,
                                      void **__stackaddr);
extern  int  pthread_attr_getstacksize(const  pthread_attr_t  * __attr,
                                        size_t * __stacksize);
extern int pthread_attr_init(pthread_attr_t * __attr);
extern int pthread_attr_setdetachstate(pthread_attr_t * __attr,
                                        int __detachstate);
extern int pthread_attr_setguardsize(pthread_attr_t * __attr,
                                        size_t __guardsize);
extern int pthread_attr_setinheritsched(pthread_attr_t * __attr,
                                        int __inherit);
extern int pthread_attr_setschedparam(pthread_attr_t * __attr,
                                        const struct sched_param * __param);
extern int pthread_attr_setschedpolicy(pthread_attr_t * __attr,
                                        int __policy);
extern int pthread_attr_setscope(pthread_attr_t * __attr, int __scope);
extern int pthread_attr_setstack(pthread_attr_t * __attr,
                                      void *__stackaddr, size_t
extern int pthread_attr_setstackaddr(pthread_attr_t *__attr,
    void *__stackaddr);
extern int pthread_attr_setstacksize(pthread_attr_t *__attr,
    size_t __stacksize);
extern int pthread_barrier_destroy(pthread_barrier_t *__barrier);
extern int pthread_barrier_init(pthread_barrier_t *__barrier,
    const pthread_barrierattr_t *__attr,
    unsigned int __count);
extern int pthread_barrier_wait(pthread_barrier_t *__barrier);
extern int pthread_barrierattr_destroy(pthread_barrierattr_t *__attr);
extern int pthread_barrierattr_getpshared(const pthread_barrierattr_t *
    __attr, int *__pshared);
extern int pthread_barrierattr_init(pthread_barrierattr_t *__attr,
    int __pshared);
extern int pthread_cancel(pthread_t __th);
extern int pthread_cond_broadcast(pthread_cond_t *__cond);
extern int pthread_cond_destroy(pthread_cond_t *__cond);
extern int pthread_cond_init(pthread_cond_t *__cond,
    const pthread_condattr_t *__cond_attr);
extern int pthread_cond_signal(pthread_cond_t *__cond);
extern int pthread_cond_timedwait(pthread_cond_t *__cond,
    pthread_mutex_t *__mutex,
    const struct timespec *__abstime);
extern int pthread_cond_wait(pthread_cond_t *__cond,
    pthread_mutex_t *__mutex);
extern int pthread_condattr_destroy(pthread_condattr_t *__attr);
extern int pthread_condattr_getclock(const pthread_condattr_t *
    attr, clockid_t *clock_id);
extern int pthread_condattr_getpshared(const pthread_condattr_t *
    __attr, int *__pshared);
extern int pthread_condattr_init(pthread_condattr_t *__attr);
extern int pthread_condattr_setclock(pthread_condattr_t *attr,
    clockid_t clock_id);
extern int pthread_condattr_setpshared(pthread_condattr_t *__attr,
    int __pshared);
extern int pthread_create(pthread_t *__newthread,
    const pthread_attr_t *__attr,
    void *(*__start_routine) (void *), void *__arg);
extern int pthread_detach(pthread_t __th);
extern int pthread_equal(pthread_t __thread1, pthread_t __thread2);
extern void pthread_exit(void *__retval);
extern int pthread_getattr_np(pthread_t __thread, pthread_attr_t *__attr);
extern int pthread_getconcurrency(void);
extern int pthread_getcpclockid(pthread_t __thread_id,
    clockid_t * __clock_id);
extern int pthread_getschedparam(pthread_t __target_thread, int *
    __policy,
    struct sched_param *__param);
extern void *pthread_getspecific(pthread_key_t __key);
extern int pthread_join(pthread_t __th, void **__thread_return);
extern int pthread_key_create(pthread_key_t *__key,
                     void (*__destr_function) (void *));
extern int pthread_mutex_consistent(pthread_mutex_t * mutex);
extern int pthread_mutex_consistent_np(pthread_mutex_t * mutex);
extern int pthread_mutex_destroy(pthread_mutex_t * mutex,
                int __prioceiling);
extern int pthread_mutex_init(pthread_mutex_t *__mutex,
                const pthread_mutexattr_t *__mutexattr);
extern int pthread_mutex_lock(pthread_mutex_t * __mutex);
extern int pthread_mutex_setprioceiling(pthread_mutex_t * __mutex,
                int __prioceiling,
                int *__old_ceiling);
extern int pthread_mutex_timedlock(pthread_mutex_t * __mutex,
                const struct timespec *__abstime);
extern int pthread_mutex_trylock(pthread_mutex_t * __mutex);
extern int pthread_mutex_unlock(pthread_mutex_t * __mutex);
extern int pthread_mutexattr_destroy(pthread_mutexattr_t * __attr);
extern int pthread_mutexattr_getprioceiling(const pthread_mutexattr_t * __attr,
                int __prioceiling);
extern int pthread_mutexattr_getprotocol(const pthread_mutexattr_t * __attr,
                int __protocol);
extern int pthread_mutexattr_getpshared(const pthread_mutexattr_t * __attr,
                int __pshared);
extern int pthread_mutexattr_getrobust(const pthread_mutexattr_t * attr,
                int __robust);
extern int pthread_mutexattr_gettype(const pthread_mutexattr_t * __attr,
                int __kind);
extern int pthread_mutexattr_init(pthread_mutexattr_t * __attr);
extern int pthread_mutexattr_setprioceiling(pthread_mutexattr_t * __attr,
                int __prioceiling);
extern int pthread_mutexattr_setprotocol(pthread_mutexattr_t * __attr,
                int __protocol);
extern int pthread_mutexattr_setpshared(pthread_mutexattr_t * __attr,
                int __pshared);
extern int pthread_mutexattr_setrobust(pthread_mutexattr_t * attr,
                int __robust);
extern int pthread_mutexattr_setrobust_np(pthread_mutexattr_t * __attr,
                int __robustness);
extern int pthread_mutexattr_settype(pthread_mutexattr_t * __attr,
                int __kind);
extern int pthread_once(pthread_once_t * __once_control,  
void (*__init_routine) (void));
extern int pthread_rwlock_destroy(pthread_rwlock_t * __rwlock);
extern int pthread_rwlock_init(pthread_rwlock_t * __rwlock,  
const pthread_rwlockattr_t * __attr);
extern int pthread_rwlock_rdlock(pthread_rwlock_t * __rwlock);
extern int pthread_rwlock_timedrdlock(pthread_rwlock_t * __rwlock,  
const struct timespec *__abstime);
extern int pthread_rwlock_timedwrlock(pthread_rwlock_t * __rwlock,  
const struct timespec *__abstime);
extern int pthread_rwlock_tryrdlock(pthread_rwlock_t * __rwlock);
extern int pthread_rwlock_trywrlock(pthread_rwlock_t * __rwlock);
extern int pthread_rwlock_unlock(pthread_rwlock_t * __rwlock);
extern int pthread_rwlock_wrlock(pthread_rwlock_t * __rwlock);
extern int pthread_rwlockattr_destroy(pthread_rwlockattr_t * __attr);
extern int pthread_rwlockattr_getkind_np(const pthread_rwlockattr_t * __attr, int *__pref);
extern int pthread_rwlockattr_getpshared(const pthread_rwlockattr_t * __attr, int *__pshared);
extern int pthread_rwlockattr_init(pthread_rwlockattr_t * __attr);
extern int pthread_rwlockattr_setkind_np(pthread_rwlockattr_t * __attr, int __pref);
extern int pthread_rwlockattr_setpshared(pthread_rwlockattr_t * __attr, int __pshared);
extern pthread_t pthread_self(void);
extern int pthread_setcancelstate(int __state, int *__oldstate);
extern int pthread_setcanceltype(int __type, int *__oldtype);
extern int pthread_setconcurrency(int __level);
extern int pthread_setschedparam(pthread_t __target_thread, int __policy,  
const struct sched_param *__param);
extern int pthread_setschedprio(pthread_t __target_thread, int __prio);
extern int pthread_setspecific(pthread_key_t __key, const void * __pointer);
extern int pthread_spin_destroy(pthread_spinlock_t * __lock);
extern int pthread_spin_init(pthread_spinlock_t * __lock, int __pshared);
extern int pthread_spin_lock(pthread_spinlock_t * __lock);
extern int pthread_spin_trylock(pthread_spinlock_t * __lock);
extern int pthread_spin_unlock(pthread_spinlock_t * __lock);
extern void pthread_testcancel(void);

14.10.3 semaphore.h

typedef union {
    char __size[__SIZEOF_SEM_T];
    long int __align;
} sem_t;

#define SEM_FAILED      ((sem_t*)0)
#define SEM_VALUE_MAX   ((int)((~0u)>>1))
extern int sem_close(sem_t * __sem);
extern int sem_destroy(sem_t * __sem);
extern int sem_getvalue(sem_t * __sem, int *__sval);
extern int sem_init(sem_t * __sem, int __pshared, unsigned int __value);
extern sem_t *sem_open(const char *__name, int __oflag, ...);
extern int sem_post(sem_t * __sem);
extern int sem_timedwait(sem_t * __sem, const struct timespec *__abstime);
extern int sem_trywait(sem_t * __sem);
extern int sem_unlink(const char *__name);
extern int sem_wait(sem_t * __sem);

14.11 Interface Definitions for libpthread

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

Other interfaces listed in Section 14.9 shall behave as described in the referenced base document.

(pthread_cleanup_pop

Name
(pthread_cleanup_pop — establish cancellation handlers

Synopsis

#include <pthread.h>
void _pthread_cleanup_pop(struct _pthread_cleanup_buffer *, int);

Description

The _pthread_cleanup_pop() function provides an implementation of the pthread_cleanup_pop() macro described in POSIX_1003.1-2008 (ISO/IEC 9945-2009).

The _pthread_cleanup_pop() function is not in the source standard; it is only in the binary standard.
_pthread_cleanup_push

Name

(pthread_cleanup_push — establish cancellation handlers)

Synopsis

#include <pthread.h>
void _pthread_cleanup_push(struct _pthread_cleanup_buffer *, void (*) (void *), void *);

Description

The _pthread_cleanup_push() function provides an implementation of the
pthread_cleanup_push() macro described in POSIX 1003.1-2008 (ISO/IEC 9945-
2009).

The pthread_cleanup_push() function is not in the source standard; it is only in
the binary standard.
**pthread_getattr_np**

**Name**

pthread_getattr_np — get thread attributes

**Synopsis**

```c
#include <pthread.h>
int pthread_getattr_np(pthread_t thread, pthread_attr_t *attr);
```

**Description**

pthread_getattr_np() fills in the thread attribute object *attr* with attribute values describing the running thread *thread*. This is useful to detect runtime changes from the values specified in the thread attributes object used to create the thread with pthread_create(). The following differences may be noted:

- The detach state, since a joinable thread may have detached itself after creation. Use pthread_attr_getdetachstate() to extract from *attr*.
- The stack size, which the implementation may align to a suitable boundary. Use pthread_attr_getstack() to extract from *attr*.
- The guard size, which the implementation may round upwards to a multiple of the page size, or ignore (i.e., treat as 0), if the application is allocating its own stack. Use pthread_attr_getguardsize() to extract from *attr*.

If the stack address attribute was not set in the thread attributes object used to create the thread, then the thread attributes object returned by pthread_getattr_np() will show the actual stack address the implementation selected for the thread. Use pthread_attr_getstack() to extract from *attr*.

The thread attributes object *attr* should be destroyed using pthread_attr_destroy() when it is no longer needed.

**Return Value**

On success, pthread_getattr_np() returns 0; on error, it returns a non-zero error number.

**Errors**

ENOMEM

Insufficient memory to complete the operation.

In addition, if *thread* refers to the main thread, then pthread_getattr_np() may also fail due to errors from various underlying calls: fopen(), if the pseudo-file containing the memory region map cannot be opened; getrlimit() if the RLIMIT_STACK resource limit it not supported.

**Notes**

This function is a GNU extension.

**See Also**

pthread_attr_destroy(). pthread_attr_getdetachstate().

pthread_attr_getguardsize(). pthread_attr_getstack(). pthread_create().
**pthread_mutex_consistent_np**

**Name**

`pthread_mutex_consistent_np` — mark state protected by robust mutex as consistent

**Synopsis**

```c
#include <pthread.h>
int pthread_mutex_consistent_np(pthread_mutex_t *__mutex);
```

**Description**

`pthread_mutex_consistent_np()` shall behave as described for `pthread_mutex_consistent()` in *POSIX 1003.1-2008 (ISO/IEC 9945-2009)*.

**pthread_mutexattr_getrobust_np,**

**pthread_mutexattr_setrobust_np**

**Name**

`pthread_mutexattr_getrobust_np`,

`pthread_mutexattr_setrobust_np` — get and set the mutex robust attribute

**Synopsis**

```c
#include <pthread.h>
int pthread_mutexattr_getrobust_np(const pthread_mutexattr_t *__attr, int *__robustness);
in int pthread_mutexattr_setrobust_np(const pthread_mutexattr_t *__attr, int __robustness);
```

**Description**

`pthread_mutexattr_setrobust_np()` shall behave as described for `pthread_mutexattr_setrobust()` in *POSIX 1003.1-2008 (ISO/IEC 9945-2009)*.

`pthread_mutexattr_getrobust_np()` shall behave as described for `pthread_mutexattr_getrobust()` in *POSIX 1003.1-2008 (ISO/IEC 9945-2009)*.

Two additional valid values are defined for `__robustness:`

- `PTHREAD_MUTEX_STALLED_NP`, which is identical to `PTHREAD_MUTEX_STALLED`
- `PTHREAD_MUTEX_ROBUST_NP`, which is identical to `PTHREAD_MUTEX_ROBUST`.

© 2015 Linux Foundation
Name

pthread_rwlockattr_getkind_np, pthread_rwlockattr_setkind_np — get/set the read-write lock kind of the thread read-write lock attribute object

Synopsis

#include <pthread.h>

int pthread_rwlockattr_getkind_np(const pthread_rwlockattr_t * attr, int * pref);

int pthread_rwlockattr_setkind_np(pthread_rwlockattr_t * attr, int * pref);

Description

The pthread_rwlockattr_setkind_np() function sets the kind of read-write lock of the thread read-write lock attribute object referred to by attr to the value specified with pref. The argument pref may be set to PTHREAD_RWLOCK_PREFER_READER_NP, PTHREAD_RWLOCK_PREFER_WRITER_NONRECURSIVE_NP, or PTHREAD_RWLOCK_PREFER_WRITER_NP. The default lock setting is PTHREAD_RWLOCK_PREFER_READER_NP. A thread may hold multiple read locks, i.e. read locks are recursive. According to The Single Unix Specification, the behavior is unspecified when a reader tries to place a lock, and there is no write lock but writers are waiting. Giving preference to the reader, as is set by default with the PTHREAD_RWLOCK_PREFER_READER_NP value implies that the reader will receive the requested lock, even if a writer is waiting. As long as there are readers the writer will be starved. Setting the kind to PTHREAD_RWLOCK_PREFER_WRITER_NONRECURSIVE_NP, avoids writer starvation as long as any read locking is not done in a recursive fashion.

The pthread_rwlockattr_getkind_np() function returns the value of the read-write lock attribute of the thread read-write lock attribute object referred to by attr in the pointer pref.

Return Value

pthread_rwlockattr_setkind_np() function returns 0 on success; on error, it returns a non-zero error number. pthread_rwlockattr_setkind_np() function always returns 0.

Errors

EINVAL

 prefetch is set to an unsupported value.

Notes

Setting the value read-write lock kind to PTHREAD_RWLOCK_PREFER_WRITER_NP, results in the same behavior as setting the value to PTHREAD_RWLOCK_PREFER_READER_NP. As long as a reader thread holds the lock the thread holding a write lock will be starved. Setting the kind value to PTHREAD_RWLOCK_PREFER_WRITER_NONRECURSIVE_NP, allows the writer to run. However, the writer may not be recursive as is implied by the name.
waitpid

Name

waitpid — wait for child process

Description

waitpid() is as specified in POSIX.1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

Need not support WCONTINED or WIFCONTINED

Implementations need not support the XSI optional functionality of WCONTINED() or WIFCONTINED().

14.12 Interfaces for libgcc_s

Table 14-55 defines the library name and shared object name for the libgcc_s library.

<table>
<thead>
<tr>
<th>Library:</th>
<th>libgcc_s</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libgcc_s.so.1</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications: [LSB] This Specification

14.12.1 Unwind Library

14.12.1.1 Interfaces for Unwind Library

An LSB conforming implementation shall provide the generic functions for Unwind Library specified in Table 14-56, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>_Unwind_Resume_or_Rethrow [LSB]</td>
<td>_Unwind_SetGR [LSB]</td>
<td>_Unwind_SetIP [LSB]</td>
<td></td>
</tr>
</tbody>
</table>

14.13 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.

© 2015 Linux Foundation
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.

### 14.13.1 unwind.h

```c
struct _Unwind_Context;
struct _Unwind_Exception;

typedef unsigned int _Unwind_Ptr __attribute__((__mode__(__pointer__)));
typedef unsigned int _Unwind_Word __attribute__((__mode__(__word__)));
typedef unsigned int _Unwind_Exception_Class __attribute__((__mode__(__DI__)));

typedef enum {
    _URC_NO_REASON = 0,
    _URC_FOREIGN_EXCEPTION_CAUGHT = 1,
    _URC_FATAL_PHASE2_ERROR = 2,
    _URC_FATAL_PHASE1_ERROR = 3,
    _URC_NORMAL_STOP = 4,
    _URC_END_OF_STACK = 5,
    _URC_HANDLER_FOUND = 6,
    _URC_INSTALL_CONTEXT = 7,
    _URC_CONTINUE_UNWIND = 8
} _Unwind_Reason_Code;

typedef void (*_Unwind_Exception_Cleanup_Fn)(_Unwind_Reason_Code, 
                                            struct _Unwind_Exception *);

struct _Unwind_Exception {
    _Unwind_Exception_Class exception_class;
    _Unwind_Exception_Cleanup_Fn exception_cleanup;
    _Unwind_Word private_1;
    _Unwind_Word private_2;
} __attribute__((__aligned__));

#define _UA_SEARCH_PHASE 1
#define _UA_END_OF_STACK 16
#define _UA_CLEANUP_PHASE 2
#define _UA_HANDLER_FRAME 4
#define _UA_FORCE_UNWIND 8

typedef int _Unwind_Action;

typedef _Unwind_Reason_Code(*_Unwind_Stop_Fn)(int version, 
                                           _Unwind_Action actions, 
                                           _Unwind_Reason_Code(*_Unwind_Stop_Fn) (_Unwind_Stop_Fn) 
                                           actions, 
                                           _Unwind_Exception* exceptionClass, 
                                           struct _Unwind_Exception* exceptionObject, 
                                           struct
```
14.14 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 14.12 shall behave as described in the referenced base document.

_Unwind_Backtrace

Name

_Unwind_Backtrace — private C++ error handling method

Synopsis

_Unwind_Reason_Code _Unwind_Backtrace(_Unwind_Trace_Fn trace, void * trace_argument);

Description

_Unwind_Backtrace() performs a stack backtrace using unwind data. The trace callback is called for every stack frame in the call chain. No cleanup actions are performed.
**_Unwind_DeleteException**

**Name**

_Unwind_DeleteException — private C++ error handling method

**Synopsis**

```c
void _Unwind_DeleteException(struct _Unwind_Exception * object);
```

**Description**

_Unwind_DeleteException() deletes the given exception object. If a given runtime resumes normal execution after catching a foreign exception, it will not know how to delete that exception. Such an exception shall be deleted by calling _Unwind_DeleteException(). This is a convenience function that calls the function pointed to by the exception_cleanup field of the exception header.

**_Unwind_FindEnclosingFunction**

**Name**

_Unwind_FindEnclosingFunction — private C++ error handling method

**Synopsis**

```c
void * _Unwind_FindEnclosingFunction(void * ip);
```

**Description**

_Unwind_FindEnclosingFunction() finds the start address of the procedure containing the specified ip or NULL if it cannot be found (for example, because the function has no unwind info).

Note that there is not necessarily a one-to-one correspondence between source level functions and procedures. Some functions do not have unwind-info and others are split into multiple procedures.
_Unwind_ForcedUnwind

Name
_Unwind_ForcedUnwind — private C++ error handling method

Synopsis
#include <unwind.h>
_Unwind_Reason_Code _Unwind_ForcedUnwind(struct _Unwind_Exception * object, _Unwind_Stop_Fn stop, void * stop_parameter);

Description
Forced unwinding is a single-phase process. stop and stop_parameter control the termination of the unwind process instead of the usual personality routine query. Stop function stop is called for each unwind frame, with the parameters described for the usual personality routine below, plus an additional stop_parameter.

Return Value
When stop identifies the destination frame, it transfers control to the user code as appropriate without returning, normally after calling _Unwind_DeleteException(). If not, then it should return an _Unwind_Reason_Code value.

If stop returns any reason code other than _URC_NO_REASON, then the stack state is indeterminate from the point of view of the caller of _Unwind_ForcedUnwind(). Rather than attempt to return, therefore, the unwind library should use the exception_cleanup entry in object, and then call abort().

_URC_NO_REASON
This is not the destination from. The unwind runtime will call frame's personality routine with the _UA_FORCE_UNWIND and _UA_CLEANUP_PHASE flag set in actions, and then unwind to the next frame and call the stop() function again.

_URC_END_OF_STACK
In order to allow _Unwind_ForcedUnwind() to perform special processing when it reaches the end of the stack, the unwind runtime will call it after the last frame is rejected, with a NULL stack pointer in the context, and the STOP() FUNCTION SHALL CATCH THIS CONDITION. IT MAY return this code if it cannot handle end-of-stack.

_URC_FATAL_PHASE2_ERROR
The stop() function may return this code for other fatal conditions like stack corruption.
_Unwind_GetCFA

Name
_Unwind_GetCFA — private C++ error handling method

Synopsis
_Unwind_Word _Unwind_GetCFA(struct _Unwind_Context * context);

Description
_Unwind_GetCFA() shall retrieve the value of the Canonical Frame Address (CFA) of
the given context.

_Unwind_GetGR

Name
_Unwind_GetGR — private C++ error handling method

Synopsis
_Unwind_Word _Unwind_GetGR(struct _Unwind_Context * context, int index);

Description
_Unwind_GetGR() returns data at index found in context. The register is identified
by its index: 0 to 31 are for the fixed registers, and 32 to 127 are for the stacked reg-
isters.
During the two phases of unwinding, only GR1 has a guaranteed value, which is the
global pointer of the frame referenced by the unwind context. If the register has its
NAT bit set, the behavior is unspecified.

_Unwind_GetIP

Name
_Unwind_GetIP — private C++ error handling method

Synopsis
_Unwind_Ptr _Unwind_GetIP(struct _Unwind_Context * context);

Description
_Unwind_GetIP() returns the instruction pointer value for the routine identified by the
unwind context.
_Unwind_GetIPInfo

**Name**

_Unwind_GetIPInfo — private C++ error handling method

**Synopsis**

_Unwind_Ptr _Unwind_GetIPInfo(struct _Unwind_Context * context, int * ip_before_insn);

**Description**

_Unwind_GetIPInfo() returns the instruction pointer value for the routine identified by the unwind context and sets ip_beforeInsn flag indicating whether that IP is before or after first not yet fully executed instruction.

_Unwind_GetLanguageSpecificData

**Name**

_Unwind_GetLanguageSpecificData — private C++ error handling method

**Synopsis**

#include <unwind.h>
_Unwind_Ptr _Unwind_GetLanguageSpecificData(struct _Unwind_Context * context);

**Description**

_Unwind_GetLanguageSpecificData() returns the address of the language specific data area for the current stack frame described by context.

_Unwind_GetRegionStart

**Name**

_Unwind_GetRegionStart — private C++ error handling method

**Synopsis**

_Unwind_Ptr _Unwind_GetRegionStart(struct _Unwind_Context * context);

**Description**

_Unwind_GetRegionStart() routine returns the address (i.e., 0) of the beginning of the procedure or code fragment described by the current unwind descriptor block.
_Unwind_RaiseException

Name
_Unwind_RaiseException — private C++ error handling method

Synopsis
_Unwind_Reason_Code _Unwind_RaiseException(struct _Unwind_Exception * object);

Description
_Unwind_RaiseException() raises an exception, passing along the given exception object, which should have its exception_class and exception_cleanup fields set. The exception object has been allocated by the language-specific runtime, and has a language-specific format, exception that it shall contain an _Unwind_Exception.

Return Value
_Unwind_RaiseException() does not return unless an error condition is found. If an error condition occurs, an _Unwind_Reason_Code is returned:

_URIC_END_OF_STACK
The unwinder encountered the end of the stack during phase one without finding a handler. The unwind runtime will not have modified the stack. The C++ runtime will normally call uncaught_exception() in this case.

_URIC_FATAL_PHASE1_ERROR
The unwinder encountered an unexpected error during phase one, because of something like stack corruption. The unwind runtime will not have modified the stack. The C++ runtime will normally call terminate() in this case.

_URIC_FATAL_PHASE2_ERROR
The unwinder encountered an unexpected error during phase two. This is usually a throw, which will call terminate().

_Unwind_Resume

Name
_Unwind_Resume — private C++ error handling method

Synopsis
void _Unwind_Resume(struct _Unwind_Exception * object);

Description
_Unwind_Resume() resumes propagation of an existing exception object. A call to this routine is inserted as the end of a landing pad that performs cleanup, but does not resume normal execution. It causes unwinding to proceed further.
__Unwind.Resume_or.Rethrow

**Name**
__Unwind.Resume_or.Rethrow — private C++ error handling method

**Synopsis**

```c
__Unwind_Reason_Code __Unwind.Resume_or.Rethrow(struct __Unwind_Exception * exception_object);
```

**Description**
If the unwind was initiated due to a forced unwind, __Unwind.Resume.or.Rethrow() shall resume that operation, else it shall re-raise the exception.

__Unwind_SetGR

**Name**
__Unwind_SetGR — private C++ error handling method

**Synopsis**

```c
void __Unwind_SetGR(struct __Unwind_Context * context, int index, uint value);
```

**Description**
__Unwind_SetGR() sets the value of the register indexed for the routine identified by the unwind context.

__Unwind_SetIP

**Name**
__Unwind_SetIP — private C++ error handling method

**Synopsis**

```c
#include <unwind.h>
void __Unwind_SetIP(struct __Unwind_Context * context, __Unwind_Ptr value);
```

**Description**
__Unwind_SetIP() sets the instruction pointer for the routine identified by the unwind context to value.

### 14.15 Interfaces for libdl

Table 14-57 defines the library name and shared object name for the libdl library

#### Table 14-57 libdl Definition

<table>
<thead>
<tr>
<th>Library:</th>
<th>libdl</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libdl.so.2</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications: [LSB] This Specification
14.15.1 Dynamic Loader

14.15.1.1 Interfaces for Dynamic Loader

An LSB conforming implementation shall provide the generic functions for Dynamic Loader specified in Table 14-58, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-58 libdl - Dynamic Loader Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>LSB</th>
<th>SUSv4</th>
</tr>
</thead>
<tbody>
<tr>
<td>dladdr</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>dlclose</td>
<td></td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>dlerror</td>
<td></td>
<td>[SUSv4]</td>
</tr>
<tr>
<td>dlopen</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>dlsym</td>
<td>[LSB]</td>
<td></td>
</tr>
<tr>
<td>dlvsym</td>
<td>[LSB]</td>
<td></td>
</tr>
</tbody>
</table>

14.16 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.

14.16.1 dlfcn.h

```c
#define RTLD_NEXT ((void *) -1l)
#define RTLD_DEFAULT ((void *) 0)
#define RTLD_LOCAL   0
#define RTLD_LAZY    0x00001
#define RTLD_NOW     0x00002
#define RTLD_NOLOAD  0x00004
#define RTLD_DEEPBIND 0x00008
#define RTLD_GLOBAL  0x00100
#define RTLD_NODELETE 0x01000

typedef struct {
    char *dli_fname;
    void *dli_fbase;
    char *dli_sname;
    void *dli_saddr;
} Dl_info;
extern int dladdr(const void *__address, Dl_info * __info);
extern int dlclose(void *__handle);
extern char *dlerror(void);
extern void *dlopen(const char *__file, int __mode);
extern void *dlsym(void *__handle, const char *__name);
extern void *dlvsym(void *handle, const char *name, const char *version);
```
14.17 Interface Definitions for libdl

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

Other interfaces listed in Section 14.15 shall behave as described in the referenced base document.

dladdr

**Name**

dladdr — find the shared object containing a given address

**Synopsis**

```
#include <dlfcn.h>

typedef struct {
    const char *dli_fname;
    void *dli_fbase;
    const char *dli_sname;
    void *dli_saddr;
```
```c
} DL_info;

int dladdr(const void * addr, DL_info * dip);
```

## Description

The `dladdr()` function shall query the dynamic linker for information about the shared object containing the address `addr`. The information shall be returned in the user supplied data structure referenced by `dip`.

The structure shall contain at least the following members:

- **dli_fname**
  
  The pathname of the shared object containing the address.

- **dli_fbase**
  
  The base address at which the shared object is mapped into the address space of the calling process.

- **dli_sname**
  
  The name of the nearest runtime symbol with value less than or equal to `addr`. Where possible, the symbol name shall be returned as it would appear in C source code.

  If no symbol with a suitable value is found, both this field and `dli_saddr` shall be set to `NULL`.

- **dli_saddr**
  
  The address of the symbol returned in `dli_sname`. This address has type "pointer to `type"", where `type` is the type of the symbol `dli_sname`.

  **Example:** If the symbol in `dli_sname` is a function, then the type of `dli_saddr` is of type "pointer to function".

The behavior of `dladdr()` is only specified in dynamically linked programs.

## Return Value

On success, `dladdr()` shall return non-zero, and the structure referenced by `dip` shall be filled in as described. Otherwise, `dladdr()` shall return zero, and the cause of the error can be fetched with `dlerror()`.

## Errors

See `dlerror()`.

## Environment

`LD_LIBRARY_PATH`

directory search-path for object files
dlopen

Name
dlopen — open dynamic object

Synopsis

```
#include <dlfcn.h>

void * dlopen(const char * filename, int flag);
```

Description

The dlopen() function shall behave as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with additional behaviors listed below.

If the file argument does not contain a <slash> character, then the system shall look for a library of that name in at least the following directories, and use the first one which is found:

- The directories specified by the DT_RPATH dynamic entry.
- The directories specified in the LD_LIBRARY_PATH environment variable (which is a colon separated list of pathnames). This step shall be skipped for setuid and setgid executable.
- A set of directories sufficient to contain the libraries specified in this standard.

Note: Traditionally, /lib and /usr/lib. This case would also cover cases in which the system used the mechanism of /etc/ld.so.conf and /etc/ld.so.cache to provide access.

Example: An application which is not linked against libm may choose to dlopen libm.

Additional flags

In addition to the available values for flag as documented in POSIX 1003.1-2008 (ISO/IEC 9945-2009), the following values may also be ORed into flag:

RTLD_NODELETE

Do not unload the library during dlclose(). Consequently, the library’s static variables are not reinitialized if the library is reloaded with dlopen() at a later time.

RTLD_NOLOAD

Do not load the library. This can be used to test if the library is already resident. dlopen() returns a NULL pointer if it is not resident; it returns the library’s handle if it is resident. This flag can also be used to promote the flags on a library that is already loaded. For example, a library that was previously loaded with RTLD_LOCAL can be reopened using RTLD_NOLOAD | RTLD_GLOBAL.

RTLD_DEEPBIND

Place the lookup scope of the symbols in this library ahead of the global scope. This means that a self-contained library will use its own symbols in preference to global symbols with the same name contained in libraries that have already been loaded.
dlsym

Name
dlsym — obtain the address of a symbol from a dlopen object

Description
dlsym() is as specified in the POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

RTLD_NEXT, RTLD_DEFAULT Required
The values RTLD_NEXT and RTLD_DEFAULT, described as reserved for future use in POSIX 1003.1-2008 (ISO/IEC 9945-2009), are required, with behavior as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009).

dlvsym

Name
dlvsym — obtain the address of a symbol from a dlopen object

Synopsis
#include <dlfcn.h>
void * dlvsym(void * handle, char * name, char * version);

Description
dlvsym() does the same as dlsym() but takes a version string as an additional argument.

14.18 Interfaces for librt

Table 14-59 defines the library name and shared object name for the librt library

Table 14-59 librt Definition

<table>
<thead>
<tr>
<th>Library:</th>
<th>librt</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>librt.so.1</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications:
[LFS] Large File Support

14.18.1 Shared Memory Objects

14.18.1.1 Interfaces for Shared Memory Objects
An LSB conforming implementation shall provide the generic functions for Shared Memory Objects specified in Table 14-60, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-60 librt - Shared Memory Objects Function Interfaces

<table>
<thead>
<tr>
<th>shm_open</th>
<th>shm_unlink</th>
</tr>
</thead>
<tbody>
<tr>
<td>[SUSv4]</td>
<td></td>
</tr>
</tbody>
</table>
14.18.2 Asynchronous I/O

14.18.2.1 Interfaces for Asynchronous I/O

An LSB conforming implementation shall provide the generic functions for Asynchronous I/O specified in Table 14-61, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-61 librt - Asynchronous I/O Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>SUSv4</th>
<th>LFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>aio_cancel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_cancel64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[LFS]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_error</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_error64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[LFS]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_fsync</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_fsync64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[LFS]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_read</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_read64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[LFS]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_return</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_return64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[LFS]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_suspend</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_suspend64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[LFS]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_write</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aio_write64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[LFS]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lio_listio(GLIBC_2.4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lio_listio64(GLIBC_C_2.4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[LFS]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

14.18.3 Clock

14.18.3.1 Interfaces for Clock

An LSB conforming implementation shall provide the generic functions for Clock specified in Table 14-62, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-62 librt - Clock Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>SUSv4</th>
<th>LFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>clock_getcpuclockid</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clock_getres</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clock_gettime</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clock_nanosleep</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

14.18.4 Timers

14.18.4.1 Interfaces for Timers

An LSB conforming implementation shall provide the generic functions for Timers specified in Table 14-63, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-63 librt - Timers Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>SUSv4</th>
<th>LFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>timer_create</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>timer_delete</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>timer_getoverrun</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>timer_gettime</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

14.18.5 Message Queues

14.18.5.1 Interfaces for Message Queues

An LSB conforming implementation shall provide the generic functions for Message Queues specified in Table 14-64, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-64 librt - Message Queues Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>SUSv4</th>
<th>LFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>mq_close(GLIBC_2.3.4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mq_getattr(GLIBC_C_2.3.4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mq_notify(GLIBC_2.3.4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mq_open(GLIBC_2.3.4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[SUSv4]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
14.19 Data Definitions for librt

This section defines global identifiers and their values that are associated with interfaces contained in librt. 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.

### 14.19.1 aio.h

```c
#define AIO_CANCELED 0
#define AIO_NOTCANCELED 1
#define AIO_ALLDONE 2

#define LIO_READ 0
#define LIO_WRITE 1
#define LIO_NOP 2
#define LIO_WAIT 0
#define LIO_NOWAIT 1

struct aiocb {
    int aio_fildes; /* File descriptor */
    int aio_lio_opcode; /* Operation to be performed */
    int aio_reqprio; /* Request priority offset */
    void *aio_buf; /* Location of buffer */
    size_t aio_nbytes; /* Length of transfer */
    struct sigevent aio_sigevent; /* Signal number and value */
    struct aiocb *__next_prio; /* internal, do not use */
    int __abs_prio; /* internal, do not use */
    int __policy; /* internal, do not use */
    int __error_code; /* internal, do not use */
    ssize_t __return_value; /* internal, do not use */
    off_t aio_offset; /* File offset */
    char __pad[sizeof(off64_t) - sizeof(off_t)];
    char __unused[32];
};

struct aiocb64 {
    int aio_fildes; /* File descriptor */
    int aio_lio_opcode; /* Operation to be performed */
    int aio_reqprio; /* Request priority offset */
    void *aio_buf; /* Location of buffer */
    size_t aio_nbytes; /* Length of transfer */
};
```
struct sigevent aio_sigevent; /* Signal number and value */
    struct aiocb *__next_prio; /* internal, do not use */
    int __abs_prio; /* internal, do not use */
    int __policy; /* internal, do not use */
    int __error_code; /* internal, do not use */
    ssize_t __return_value; /* internal, do not use */
    off64_t aio_offset; /* File offset */
    char __unused[32];
};

extern int aio_cancel(int fildes, struct aiocb *aiocbp);
extern int aio_cancel64(int fildes, struct aiocb64 *aiocbp);
extern int aio_error(struct aiocb *aiocbp);
extern int aio_error64(struct aiocb64 *aiocbp);
extern int aio_fsync(int operation, struct aiocb *aiocbp);
extern int aio_fsync64(int operation, struct aiocb64 *aiocbp);
extern int aio_read(struct aiocb *aiocbp);
extern int aio_read64(struct aiocb64 *aiocbp);
extern int aio_return(struct aiocb *aiocbp);
extern int aio_return64(struct aiocb64 *aiocbp);
extern int aio_suspend(struct aiocb *list[], int nent, 
        struct timespec *timeout);
extern int aio_suspend64(struct aiocb64 *list[], int nent, 
        struct timespec *timeout);
extern int aio_write(struct aiocb *aiocbp);
extern int aio_write64(struct aiocb64 *aiocbp);
extern int lio_listio(int mode, struct aiocb *list[], int nent, 
        struct sigevent *sig);
extern int lio_listio64(int mode, struct aiocb64 *list[], int nent, 
        struct sigevent *sig);

14.19.2 mqueue.h

typedef int mqd_t;
struct mq_attr {
    long int mq_flags;
    long int mq_maxmsg;
    long int mq_msgsize;
    long int mq_curmsgs;
    long int __pad[4];
};

extern int mq_close(mqd_t __mqdes);
extern int mq_getattr(mqd_t __mqdes, struct mq_attr *__mqstat);
extern int mq_notify(mqd_t __mqdes, const struct sigevent *__notification);
extern mqd_t mq_open(const char *__name, int __oflag, ...);
extern ssize_t mq_receive(mqd_t __mqdes, char *__msg_ptr, size_t __msg_len, 
        unsigned int *__msg_prio);
extern int mq_send(mqd_t __mqdes, const char *__msg_ptr, size_t __msg_len, 
        unsigned int __msg_prio);
extern int mq_setattr(mqd_t __mqdes, const struct mq_attr *__mqstat, 
        struct mq_attr *__omqstat);
extern ssize_t mq_timedreceive(mqd_t __mqdes, char *__msg_ptr, size_t __msg_len, 
        unsigned int *__msg_prio, 
        const struct timespec *__abs_timeout);
extern int mq_timedsend(mqd_t __mqdes, const char *__msg_ptr, size_t __msg_len, 
        unsigned int __msg_prio, 
        const struct timespec *__abs_timeout);
14 Base Libraries

extern int mq_unlink(const char *name);

14.20 Interfaces for libcrypt

Table 14-65 defines the library name and shared object name for the libcrypt library

<table>
<thead>
<tr>
<th>Library:</th>
<th>libcrypt</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libcrypt.so.1</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications:
[LSB] This Specification

14.20.1 Encryption

14.20.1.1 Interfaces for Encryption

An LSB conforming implementation shall provide the generic functions for Encryption specified in Table 14-66, with the full mandatory functionality as described in the referenced underlying specification.

Table 14-66 libcrypt - Encryption Function Interfaces

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>setkey [SUSv4]</td>
<td>setkey_r [LSB]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

14.21 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.

14.21.1 crypt.h

```c
struct crypt_data {
    char keysched[128];
    char sb0[32768];
    char sb1[32768];
    char sb2[32768];
    char sb3[32768];
    char crypt_3_buf[14];
    char current_salt[2];
    long int current_saltbits;
    int direction;
    int initialized;
};
extern char *crypt_r(const char *key, const char *salt,
```
14.22 Interface Definitions for libcrypt

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

Other interfaces listed in Section 14.20 shall behave as described in the referenced base document.

crypt_r

Name

crypt_r — Cryptographic string encoding function

Synopsis

#include <crypt.h>
char * crypt_r(const char *key, const char *salt, struct crypt_data *data);

Description

The crypt_r() function is a re-entrant version of the crypt() function. crypt_r() shall behave as specified for crypt() in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with an additional parameter, a pointer to a structure which is used to store result data and bookkeeping information.

The caller should set the initialized field of the crypt_data structure to zero before the first call to crypt_r().

Notes

INSERT TEXT HERE

See Also

crypt(), setkey_r(), encrypt_r().
**encrypt_r**

**Name**

`encrypt_r` — Cryptographic encoding function

**Synopsis**

```c
#include <crypt.h>
void encrypt_r(const char * block, int edflag, struct crypt_data * data);
```

**Description**

The `encrypt_r()` function is a re-entrant version of the `encrypt()` function. `encrypt_r()` shall behave as specified for `encrypt()` in [POSIX 1003.1-2008 (ISO/IEC 9945-2009)](https://pubs.opengroup.org/onlinepubs/9699919799/), but with an additional parameter, a pointer to a structure which is used to store result data and bookkeeping information.

**Notes**

INSERT TEXT HERE

**See Also**

`encrypt()`, `crypt_r()`, `setkey_r()`.

---

**setkey_r**

**Name**

`setkey_r` — Set cryptographic encoding key

**Synopsis**

```c
#include <crypt.h>
void setkey_r(const char * key, struct crypt_data * data);
```

**Description**

The `setkey_r()` function is a re-entrant version of the `setkey()` function. `setkey_r()` shall behave as specified for `setkey()` in [POSIX 1003.1-2008 (ISO/IEC 9945-2009)](https://pubs.opengroup.org/onlinepubs/9699919799/), but with an additional parameter, a pointer to a structure which is used to store result data and bookkeeping information.

The caller should set the `initialized` field of the `crypt_data` structure to zero before the first call to `setkey_r()`.

**Notes**

INSERT TEXT HERE

**See Also**

`setkey()`, `crypt_r()`, `encrypt_r()`.

---

### 14.23 Interfaces for libpam

Table 14-67 defines the library name and shared object name for the libpam library
The Pluggable Authentication Module (PAM) interfaces allow applications to request authentication via a system administrator defined mechanism, known as a *service*.

A single service name, other, shall always be present. The behavior of this service shall be determined by the system administrator. Additional service names may also exist.

**Note:** Future versions of this specification might define additional service names.

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

- [LSB] *This Specification*
- [PAM] *PAM*

### 14.23.1 Pluggable Authentication API

#### 14.23.1.1 Interfaces for Pluggable Authentication API

An LSB conforming implementation shall provide the generic functions for Pluggable Authentication API specified in *Table 14-68*, with the full mandatory functionality as described in the referenced underlying specification.

### 14.24 Data Definitions for libpam

This section defines global identifiers and their values that are associated with interfaces contained in libpam. 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.
14.24.1 security/_pam_types.h

typedef struct pam_handle pam_handle_t;
structure pam_message {
    int msg_style;
    const char *msg;
};
structure pam_response {
    char *resp;
    int resp_retcode; /* currently un-used, zero expected */
};
structure pam_conv {
    int (*conv) (int num_msg, const struct pam_message * *msg,
                 struct pam_response * *resp, void *appdata_ptr);
    void *appdata_ptr;
};
#define PAM_PROMPT_ECHO_OFF 1
#define PAM_PROMPT_ECHO_ON 2
#define PAM_ERROR_MSG 3
#define PAM_TEXT_INFO 4
#define PAM_SERVICE 1 /* The service name */
#define PAM_USER 2 /* The user name */
#define PAM_TTY 3 /* The tty name */
#define PAM_RHOST 4 /* The remote host name */
#define PAM_CONV 5 /* The pam_conv structure */
#define PAM_RUSER 8 /* The remote user name */
#define PAM_USER_PROMPT 9 /* the prompt for getting a username */
#define PAM_SUCCESS 0 /* Successful function return */
#define PAM_OPEN_ERR 1 /* dlopen() failure */
#define PAM_USER_UNKNOWN 10 /* User not known to the underlying authenticaiton module */
#define PAM_MAXTRIES 11 /* An authentication service has maintained a retry count which */
#define PAM_NEW_AUTHTOK_REQD 12 /* New authentication token required */
#define PAM_ACCT_EXPIRED 13 /* User account has expired */
#define PAM_SESSION_ERR 14 /* Can not make/remove an entry for the specified session */
#define PAM_CRED_UNAVAIL 15 /* Underlying authentication service can not retrieve user cred */
#define PAM_CRED_EXPIRED 16 /* User credentials expired */
#define PAM_CRED_ERR 17 /* Failure setting user credentials */
#define PAM_CONV_ERR 19 /* Conversation error */
#define PAM_SYMBOL_ERR 2 /* Symbol not found */
#define PAM_AUTHTOK_ERR 20 /* Authentication token manipulation error */
#define PAM_AUTHTOK_RECOVER_ERR 21 /* Authentication information cannot be recovered */
#define PAM_AUTHTOK_LOCK_BUSY 22 /* Authentication token lock busy */
#define PAM_AUTHTOK_DISABLE_AGING 23 /* Authentication token aging disabled */
#define PAM_TRY_AGAIN 24 /* Preliminary check by password service */
#define PAM_ABORT 26 /* Critical error (module fail now request) */
#define PAM_AUTHTOK_EXPIRED 27 /* user's authentication token has expired */
#define PAM_BAD_ITEM 29 /* Bad item passed to pam_*_item() */
#define PAM_SERVICE_ERR 3 /* Error in service module */
#define PAM_SYSTEM_ERR 4 /* System error */
#define PAM_BUF_ERR 5 /* Memory buffer error */
#define PAM_PERM_DENIED 6 /* Permission denied */
#define PAM_AUTH_ERR 7 /* Authentication failure */
#define PAM_CRED_INSUFFICIENT 8 /* Can not access authentication data due to insufficient crede */
#define PAM_AUTHINFO_UNAVAIL 9 /* Underlying authentication service can not retrieve authentic */

#define PAM_DISALLOW_NULL_AUTHTOK 0x0001U
#define PAM_ESTABLISH_CRED 0x0002U /* Set user credentials for an authentication service */
#define PAM_DELETE_CRED 0x0004U /* Delete user credentials associated with an authentication service */
#define PAM_REINITIALIZE_CRED 0x0008U /* Reinitialize user credentials */
#define PAM_REFRESH_CRED 0x0010U /* Extend lifetime of user credentials */
#define PAM_CHANGE_EXPIRED_AUTHTOK 0x0020U /* Extend lifetime of user credentials */
#define PAM_SILENT 0x8000U /* Authentication service should not generate any messages */

extern int pam_fail_delay(pam_handle_t *, unsigned int);
extern int pam_set_get_item(const pam_handle_t *, int, const void *);
extern char *pam_getenv(const pam_handle_t *, const char *);
extern char **pam_getenvlist(pam_handle_t *);
extern int pam_putenv(pam_handle_t *, const char *);
extern int pam_set_item(pam_handle_t *, int, const void *);
extern const char *pam_strerror(pam_handle_t *, int);

extern int pam_acct_mgmt(pam_handle_t *, int);
extern int pam_authenticate(pam_handle_t *, int);
extern int pam_chauthtok(pam_handle_t *, int);
extern int pam_close_session(pam_handle_t *, int);
extern int pam_end(pam_handle_t *, int);
extern int pam_open_session(pam_handle_t *, int);
extern int pam_setcred(pam_handle_t *, int);
extern int pam_start(const char *, const char *, const struct pam_conv *, pam_handle_t * *);

14.25 Interface Definitions for libpam

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

© 2015 Linux Foundation
source standard.
Other interfaces listed in Section 14.23 shall behave as described in the referenced base document.

pam_acct_mgmt

Name
pam_acct_mgmt — establish the status of a user's account

Synopsis
#include <security/pam_appl.h>
int pam_acct_mgmt(pam_handle_t * pamh, int flags);

Description
pam_acct_mgmt() establishes the account's usability and the user's accessibility to the system. It is typically called after the user has been authenticated.

flags may be specified as any valid flag (namely, one of those applicable to the flags argument of pam_authenticate()). Additionally, the value of flags may be logically or'd with PAM_SILENT.

Return Value
PAM_SUCCESS
Success.

PAM_NEW_AUTHTOK_REQD
User is valid, but user's authentication token has expired. The correct response to this return-value is to require that the user satisfy the pam_chauthtok() function before obtaining service. It may not be possible for an application to do this. In such a case, the user should be denied access until the account password is updated.

PAM_ACCT_EXPIRED
User is no longer permitted access to the system.

PAM_AUTH_ERR
Authentication error.

PAM_PERM_DENIED
User is not permitted to gain access at this time.

PAM_USER_UNKNOWN
User is not known to a module's account management component.

Note: Errors may be translated to text with pam_strerror().
# pam_authenticate

## Name

`pam_authenticate` — authenticate the user

## Synopsis

```c
#include <security/pam_appl.h>
int pam_authenticate(pam_handle_t * pamh, int flags);
```

## Description

`pam_authenticate()` serves as an interface to the authentication mechanisms of the loaded modules.

`flags` is an optional parameter that may be specified by the following value:

- `PAM_DISALLOW_NULL_AUTHTOK`
  
  Instruct the authentication modules to return `PAM_AUTH_ERR` if the user does not have a registered authorization token.

Additionally, the value of `flags` may be logically or'd with `PAM_SILENT`.

The process may need to be privileged in order to successfully call this function.

## Return Value

- `PAM_SUCCESS`
  
  Success.

- `PAM_AUTH_ERR`
  
  User was not authenticated or process did not have sufficient privileges to perform authentication.

- `PAM_CRED_INSUFFICIENT`
  
  Application does not have sufficient credentials to authenticate the user.

- `PAM_AUTHINFO_UNAVAIL`
  
  Modules were not able to access the authentication information. This might be due to a network or hardware failure, etc.

- `PAM_USER_UNKNOWN`
  
  Supplied username is not known to the authentication service.

- `PAM_MAXTRIES`
  
  One or more authentication modules has reached its limit of tries authenticating the user. Do not try again.

- `PAM_ABORT`
  
  One or more authentication modules failed to load.

**Note:** Errors may be translated to text with `pam_strerror()`. 
pam_chauthtok

Name
pam_chauthtok — change the authentication token for a given user

Synopsis
#include <security/pam_appl.h>
int pam_chauthtok(pam_handle_t * pamh, const int flags);

Description
pam_chauthtok() is used to change the authentication token for a given user as indicated by the state associated with the handle pamh.

flags is an optional parameter that may be specified by the following value:

PAM_CHANGE_EXPIRED_AUTHTOK
User's authentication token should only be changed if it has expired.

Additionally, the value of flags may be logically or'd with PAM_SILENT.

RETURN VALUE
PAM_SUCCESS
Success.

PAM_AUTHTOK_ERR
A module was unable to obtain the new authentication token.

PAM_AUTHTOK_RECOVER_ERR
A module was unable to obtain the old authentication token.

PAM_AUTHTOK_LOCK_BUSY
One or more modules were unable to change the authentication token since it is currently locked.

PAM_AUTHTOK_DISABLE_AGING
Authentication token aging has been disabled for at least one of the modules.

PAM_PERM_DENIED
Permission denied.

PAM_TRY_AGAIN
Not all modules were in a position to update the authentication token(s). In such a case, none of the user's authentication tokens are updated.

PAM_USER_UNKNOWN
User is not known to the authentication token changing service.

Note: Errors may be translated to text with pam_strerror().
pam_close_session

Name
pam_close_session — indicate that an authenticated session has ended

Synopsis
#include <security/pam_appl.h>
int pam_close_session(pam_handle_t * pamh, int flags);

Description
pam_close_session() is used to indicate that an authenticated session has ended. It is used to inform the module that the user is exiting a session. It should be possible for the PAM library to open a session and close the same session from different applications.

flags may have the value PAM_SILENT to indicate that no output should be generated as a result of this function call.

Return Value
PAM_SUCCESS
Success.
PAM_SESSION_ERR
One of the required loaded modules was unable to close a session for the user.

Note: Errors may be translated to text with pam_strerror().

pam_end

Name
pam_end — terminate the use of the PAM library

Synopsis
#include <security/pam_appl.h>
int pam_end(pam_handle_t * pamh, int pam_status);

Description
pam_end() terminates use of the PAM library. On success, the contents of *pamh are no longer valid, and all memory associated with it is invalid.

Normally, pam_status is passed the value PAM_SUCCESS, but in the event of an unsuccessful service application, the appropriate PAM error return value should be used.

Return Value
PAM_SUCCESS
Success.

Note: Errors may be translated to text with pam_strerror().
pam_fail_delay

Name
pam_fail_delay — specify delay time to use on authentication error

Synopsis
#include <security/pam_appl.h>
int pam_fail_delay(pam_handle_t * pamh, unsigned int micro_sec);

Description
pam_fail_delay() specifies the minimum delay for the PAM library to use when an authentication error occurs. The actual delay can vary by as much at 25%. If this function is called multiple times, the longest time specified by any of the call will be used.

The delay is invoked if an authentication error occurs during the pam_authenticate() or pam_chauthtok() function calls.

Independent of the success of pam_authenticate() or pam_chauthtok(), the delay time is reset to its default value of 0 when the PAM library returns control to the application from these two functions.

Return Value
PAM_SUCCESS

Success.

Note: Errors may be translated to text with pam_strerror().
pam_get_item

Name

pam_get_item — obtain the value of the indicated item.

Synopsis

#include <security/pam_appl.h>
int pam_get_item(const pam_handle_t * pamh, int item_type, const void ** item);

Description

pam_get_item() obtains the value of the indicated item_type. The possible values of item_type are the same as listed for pam_set_item().

On success, item contains a pointer to the value of the corresponding item. Note that this is a pointer to the actual data and should not be free()’d or over-written.

Return Value

PAM_SUCCESS
Success.

PAM_PERM_DENIED
Application passed a NULL pointer for item.

PAM_BAD_ITEM
Application attempted to get an undefined item.

Note: Errors may be translated to text with pam_strerror().

pam_getenv

Name

pam_getenv — get a PAM environment variable

Synopsis

#include <security/pam_appl.h>
const char * pam_getenv(const pam_handle_t * pamh, const char * name);

Description

The pam_getenv() function shall search the environment associated with the PAM handle pamh for the environment variable name. If the specified environment variable cannot be found, a null pointer shall be returned. The application shall ensure that it does not modify the string pointed to by the pam_getenv() function.

Return Value

On success, pam_getenv() returns a pointer to a string of the form name=value.
pam_getenvlist

Name

pam_getenvlist — returns a pointer to the complete PAM environment.

Synopsis

#include <security/pam_appl.h>
char * const * pam_getenvlist(pam_handle_t * pamh);

Description

pam_getenvlist() returns a pointer to the complete PAM environment. This pointer points to an array of pointers to NUL-terminated strings and must be terminated by a NULL pointer. Each string has the form "name=value".

The PAM library module allocates memory for the returned value and the associated strings. The calling application is responsible for freeing this memory.

Return Value

pam_getenvlist() returns an array of string pointers containing the PAM environment. On error, NULL is returned.

pam_open_session

Name

pam_open_session — indicate session has started

Synopsis

#include <security/pam_appl.h>
int pam_open_session(pam_handle_t * pamh, int flags);

Description

The pam_open_session() function is used to indicate that an authenticated session has begun, after the user has been identified (see pam_authenticate()) and, if necessary, granted credentials (see pam_setcred()). It is used to inform the module that the user is currently in a session. It should be possible for the PAM library to open a session and close the same session from different applications.

flags may have the value PAM_SILENT to indicate that no output be generated as a result of this function call.

Return Value

PAM_SUCCESS

Success.

PAM_SESSION_ERR

One of the loaded modules was unable to open a session for the user.

Note: Errors may be translated to text with pam_strerror().
pam_putenv

Name
pam_putenv — Add, replace or delete a PAM environment variable

Synopsis
#include <security/pam_appl.h>
int pam_putenv(const pam_handle_t * pamh, const char * name_value);

Description
The pam_putenv() function shall modify the environment list associated with pamh. If name_value contains an '=' character, the characters to the left of the first '=' character represent the name, and the remaining characters after the '=' represent the value.

If the name environment variable exists in the environment associated with pamh, it shall be modified to have the value value. Otherwise, the name shall be added to the environment associated with pamh with the value value.

If there is no '=' character in name_value, the variable in the environment associated with pamh named name_value shall be deleted.

Return Value
On success, the pam_putenv() function shall return PAM_SUCCESS. Otherwise the return value indicates the error:

PAM_PERM_DENIED
    The name_value argument is a null pointer.

PAM_BAD_ITEM
    The PAM environment variable named name_value does not exist and therefore cannot be deleted.

PAM_ABORT
    The PAM handle identified by pamh is corrupt.

PAM_BUF_ERR
    Memory buffer error.
pam_set_item

Name
pam_set_item — (re)set the value of an item.

Synopsis
#include <security/pam_appl.h>
int pam_set_item(pam_handle_t * pamh, int item_type, const void * item);

Description
pam_set_item() (re)sets the value of one of the following item_types:

PAM_SERVICE
  service name
PAM_USER
  user name
PAM_TTY
  terminal name
  The value for a device file should include the /dev/ prefix. The value for graphical, X-based, applications should be the $DISPLAY variable.

PAM_RHOST
  remote host name
PAM_CONV
  conversation structure
PAM_RUSER
  remote user name
PAM_USER_PROMPT
  string to be used when prompting for a user's name
  The default value for this string is Please enter username: .

For all item_types other than PAM_CONV, item is a pointer to a NULL-terminated character string. In the case of PAM_CONV, item points to an initialized pam_conv structure.

Return Value
PAM_SUCCESS
  Success.

PAM_PERM_DENIED
  An attempt was made to replace the conversation structure with a NULL value.

PAM_BUF_ERR
  Function ran out of memory making a copy of the item.
PAM_BAD_ITEM

Application attempted to set an undefined item.

Note: Errors may be translated to text with pam_strerror().

pam_setcred

Name

pam_setcred — set the module-specific credentials of the user

Synopsis

#include <security/pam_appl.h>
extern int pam_setcred(pam_handle_t * pamh, int flags);

Description

pam_setcred() sets the module-specific credentials of the user. It is usually called after the user has been authenticated, after the account management function has been called and after a session has been opened for the user.

flags maybe specified from among the following values:

PAM_ESTABLISH_CRED
    set credentials for the authentication service

PAM_DELETE_CRED
    delete credentials associated with the authentication service

PAM_REINITIALIZE_CRED
    reinitialize the user credentials

PAM_REFRESH_CRED
    extend lifetime of the user credentials

Additionally, the value of flags may be logically or'd with PAM_SILENT.

Return Value

PAM_SUCCESS
    Success.

PAM_CRED_UNAVAIL
    Module cannot retrieve the user's credentials.

PAM_CRED_EXPIRED
    User's credentials have expired.

PAM_USER_UNKNOWN
    User is not known to an authentication module.

PAM_CRED_ERR
    Module was unable to set the credentials of the user.

Note: Errors may be translated to text with pam_strerror().
pam_start

Name
pam_start — initialize the PAM library

Synopsis
#include <security/pam_appl.h>
int pam_start(const char *service_name, const char *user,
const struct pam_conv *pam_conversation, pam_handle_t **pamh);

Description
pam_start() is used to initialize the PAM library. It must be called prior to any other
usage of the PAM library. On success, *pamh becomes a handle that provides continuity
for successive calls to the PAM library. pam_start() expects arguments as follows:
the service_name of the program, the username of the individual to be authenticated,
a pointer to an application-supplied pam_conv structure, and a pointer to a
pam_handle_t pointer.

An application must provide the conversation function used for direct communication
between a loaded module and the application. The application also typically provides a
means for the module to prompt the user for a password, etc.

The structure, pam_conv, is defined to be,

struct pam_conv {
    int (*conv)(int num_msg,
                const struct pam_message **msg,
                struct pam_response **resp,
                void *appdata_ptr);
    void *appdata_ptr;
}
It is initialized by the application before it is passed to the library. The contents of this structure are attached to the *pamh handle. The point of this argument is to provide a mechanism for any loaded module to interact directly with the application program; this is why it is called a conversation structure.

When a module calls the referenced conv() function, appdata_ptr is set to the second element of this structure.

The other arguments of a call to conv() concern the information exchanged by module and application. num_msg holds the length of the array of pointers passed via msg. On success, the pointer resp points to an array of num_msg pam_response structures, holding the application-supplied text. Note that resp is a struct pam_response array and not an array of pointers.

**Return Value**

PAM_SUCCESS

Success.

PAM_BUF_ERR

Memory allocation error.

PAM_ABORT

Internal failure.

**ERRORS**

May be translated to text with pam_strerror().

**pam_strerror**

**Name**

pam_strerror — returns a string describing the PAM error

**Synopsis**

```c
#include <security/pam_appl.h>
const char * pam_strerror(pam_handle_t * pamh, int errnum);
```

**Description**

pam_strerror() returns a string describing the PAM error associated with errnum.

**Return Value**

On success, this function returns a description of the indicated error. The application should not free or modify this string. Otherwise, a string indicating that the error is unknown shall be returned. It is unspecified whether or not the string returned is translated according to the setting of LC_MESSAGES.
IV Utility Libraries
15 Utility Libraries

15.1 Introduction

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.

- libz
- libncurses
- libncursesw
- libutil

The structure of the definitions for these libraries follows the same model as used for Base Libraries.

15.2 Interfaces for libz

Table 15-1 defines the library name and shared object name for the libz library.

<table>
<thead>
<tr>
<th>Library:</th>
<th>libz</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libz.so.1</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications: [LSB] This Specification

15.2.1 Compression Library

15.2.1.1 Interfaces for Compression Library

An LSB conforming implementation shall provide the generic functions for Compression Library specified in Table 15-2, with the full mandatory functionality as described in the referenced underlying specification.

The behavior of the interfaces in this library is specified by the following specifications: [LSB] This Specification

<table>
<thead>
<tr>
<th>Function</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adler32</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>compress</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>compress2</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>compressBound</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>crc32</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>deflate</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>deflateBound</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>deflateCopy</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>deflateEnd</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>deflateInit</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>deflateInit2</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>deflateParams</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>deflatePrime</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>deflateReset</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>deflateSetDictionary</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>get_crc_table</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzclearerr</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzclose</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzdopen</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzeof</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzerror</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzflush</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzgetc</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzdgets</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzopen</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzprintf</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzputc</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzputs</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzread</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzrewind</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzseek</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzsetparams</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gztell</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>gzwrite</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>inflate</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>inflateBack</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>inflateBackInit</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>inflateCopy</td>
<td>ZLIB_1.2.0</td>
</tr>
<tr>
<td>inflateEnd</td>
<td>ZLIB_1.2.0</td>
</tr>
</tbody>
</table>
15.3 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.

15.3.1 zconf.h

```c
#define ZEXPORT
#define ZEXPORTVA
#define OF(args)        args
#define ZEXTERN extern
```

15.3.2 zlib.h

```c
#define ZLIB_VERSION    "1.2.2"
#define Z_NULL  0
#define MAX_WBITS       15      /* 32K LZ77 window */
#define MAX_MEM_LEVEL   9       /* Maximum value for memLevel in deflateInit2 */
#define deflateInit2(strm,level,method,windowBits,memLevel,strategy)    
    deflateInit2_((strm),(level),(method),(windowBits),(memLevel),(strategy),ZLIB_VERSION,sizeof(z_stream))
#define deflateInit(strm,level) 
    deflateInit_((strm), (level),       ZLIB_VERSION, sizeof(z_stream))
#define inflateInit2(strm,windowBits)   
    inflateInit2_((strm), (windowBits), ZLIB_VERSION, sizeof(z_stream))
#define inflateInit(strm)       
    inflateInit_((strm),                ZLIB_VERSION, sizeof(z_stream))
#define inflateBackInit(strm, windowBits, window)       
    inflateBackInit_((strm), (windowBits), (window), ZLIB_VERSION, sizeof(z_stream))
```
typedef void *voidpf;
typedef unsigned int uInt;
typedef unsigned long int uLong;
typedef uLong uLongf;
typedef void *voidp;
typedef unsigned char Byte;
typedef off_t z_off_t;
typedef void *const voidpc;

typedef voidpf(*alloc_func) (voidpf opaque, uInt items, uInt size);
typedef void (*free_func) (voidpf opaque, voidpf address);
struct internal_state {
    int dummy;
};
typedef Byte Bytef;
typedef uInt uIntf;
typedef unsigned int (*in_func) (void *, unsigned char **);
typedef int (*out_func) (void *, unsigned char *, unsigned int);

typedef struct z_stream_s {
    Bytef *next_in;             /* next input byte */
    uInt avail_in;              /* number of bytes available at
   next_in */
    uLong total_in;             /* total nb of input bytes read
   so far */
    Bytef *next_out;            /* next output byte should be put
   there */
    uInt avail_out;             /* remaining free space at
   next_out */
    uLong total_out;            /* total nb of bytes output so
   far */
    char *msg;                  /* last error message, NULL if no
   error */
    struct internal_state *state;       /* not visible by
   applications */
    alloc_func zalloc;          /* used to allocate the internal
   state */
    free_func zfree;            /* used to free the internal
   state */
    voidpf opaque;              /* private data object passed to
   zalloc and zfree */
    int data_type;              /* best guess about the data
   type: ascii or binary */
    uLong adler;                /* adler32 value of the
   uncompressed data */
    uLong reserved;             /* reserved for future use */
} z_stream;

typedef z_stream *z_streamp;
typedef voidp gzFile;

#define Z_NO_FLUSH      0
#define Z_PARTIAL_FLUSH 1
#define Z_SYNC_FLUSH    2
#define Z_FULL_FLUSH    3
#define Z_FINISH        4
#define Z_BLOCK 5
#define Z_ERRNO (-1)
#define Z_STREAM_ERROR (-2)
#define Z_DATA_ERROR   (-3)
#define Z_MEM_ERROR    (-4)
#define Z_BUF_ERROR     (-5)
#define Z_VERSION_ERROR (-6)
15 Utility Libraries

LSB Core - Generic 5.0

#define Z_OK 0
#define Z_STREAM_END 1
#define Z_NEED_DICT 2
#define Z_DEFAULT_COMPRESSION (-1)
#define Z_NO_COMPRESSION 0
#define Z_BEST_SPEED 1
#define Z_BEST_COMPRESSION 9
#define Z_DEFAULT_STRATEGY 0
#define Z_FILTERED 1
#define Z_HUFFMAN_ONLY 2
#define Z_BINARY 0
#define Z_ASCII 1
#define Z_UNKNOWN 2
#define Z_DEFLATED 8

extern uLong adler32(uLong adler, const Bytef * buf, uInt len);
extern int compress(Bytef * dest, uLongf * destLen, const Bytef * source,
                     uLong sourceLen);
extern int compress2(Bytef * dest, uLongf * destLen, const Bytef * source,
                      uLong sourceLen, int level);
extern uLong compressBound(uLong sourceLen);
extern uLong crc32(uLong crc, const Bytef * buf, uInt len);
extern int deflate(z_streamp strm, int flush);
extern uLong deflateBound(z_streamp strm, uLong sourceLen);
extern int deflateCopy(z_streamp dest, z_streamp source);
extern int deflateEnd(z_streamp strm);
extern int deflateInit2_(z_streamp strm, int level, int method,
                         int windowBits, int memLevel, int strategy,
                         const char *version, int stream_size);
extern int deflateInit_(z_streamp strm, int level, const char *version,
                        int stream_size);
extern int deflateParams(z_streamp strm, int level, int strategy);
extern int deflatePrime(z_streamp strm, int bits, int value);
extern int deflateReset(z_streamp strm);
extern int deflateSetDictionary(z_streamp strm, const Bytef * dictionary,
                                 uInt dictLength);
extern const uLongf *get_crc_table(void);
extern void gzclearerr(gzFile file);
extern int gzclose(gzFile file);
extern gzFile gzdopen(int fd, const char *mode);
extern int gzeof(gzFile file);
extern const char *gzerror(gzFile file, int *errnum);
extern int gzflush(gzFile file, int flush);
extern char *gzgets(gzFile file, char *buf, int len);
extern gzFile gzopen(const char *path, const char *mode);
extern int gzprintf(gzFile file, const char *format, ...);
extern int gzputc(gzFile file, int c);
extern int gzputs(gzFile file, const char *s);
extern int gzread(gzFile file, voidp buf, unsigned int len);
extern int gzrewind(gzFile file);
extern int gztell(gzFile file);
extern int gzwrite(gzFile file, voidp buf, unsigned int len);
extern int inflate(z_streamp strm, int flush);
15.4 Interface Definitions for libz

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

Other interfaces listed in Section 15.2 shall behave as described in the referenced base document.

adler32

Name
adler32 — compute Adler 32 Checksum

Synopsis

#include <zlib.h>
uLong adler32(uLong adler, const Bytef * buf, uInt len);

Description

The adler32() function shall compute a running Adler-32 checksum (as described in RFC 1950: ZLIB Compressed Data Format Specication). On entry, adler is the previous value for the checksum, and buf shall point to an array of len bytes of data to be added to this checksum. The adler32() function shall return the new checksum.

If buf is NULL (or Z_NULL), adler32() shall return the initial checksum.

Return Value

The adler32() function shall return the new checksum value.

Errors

None defined.

Application Usage (informative)
The following code fragment demonstrates typical usage of the `adler32()` function:

```c
uLong adler = adler32(0L, Z_NULL, 0);
while (read_buffer(buffer, length) != EOF) {
    adler = adler32(adler, buffer, length);
}  
if (adler != original_adler) error();
```

**compress**

**Name**

`compress` — compress data

**Synopsis**

```c
#include <zlib.h>
int compress(Bytef * dest, uLongf * destLen, const Bytef * source, uLong sourceLen);
```

**Description**

The `compress()` function shall attempt to compress `sourceLen` bytes of data in the buffer `source`, placing the result in the buffer `dest`.

On entry, `destLen` should point to a value describing the size of the `dest` buffer. The application should ensure that this value be at least `(sourceLen × 1.001) + 12`. On successful exit, the variable referenced by `destLen` shall be updated to hold the length of compressed data in `dest`.

The `compress()` function is equivalent to `compress2()` with a `level` of `Z_DEFAULT_COMPRESSION`.

**Return Value**

On success, `compress()` shall return `Z_OK`. Otherwise, `compress()` shall return a value to indicate the error.

**Errors**

On error, `compress()` shall return a value as described below:

`Z_BUF_ERROR`

The buffer `dest` was not large enough to hold the compressed data.

`Z_MEM_ERROR`

Insufficient memory.
compress2

Name
compress2 — compress data at a specified level

Synopsis
#include <zlib.h>
int compress2(Bytef * dest, uLongf * destLen, const Bytef * source, uLong sourceLen, int level);

Description
The compress2() function shall attempt to compress sourceLen bytes of data in the buffer source, placing the result in the buffer dest, at the level described by level. The level supplied shall be a value between 0 and 9, or the value Z_DEFAULT_COMPRESSION. A level of 1 requests the highest speed, while a level of 9 requests the highest compression. A level of 0 indicates that no compression should be used, and the output shall be the same as the input.

On entry, destLen should point to a value describing the size of the dest buffer. The application should ensure that this value be at least (sourceLen × 1.001) + 12. On successful exit, the variable referenced by destLen shall be updated to hold the length of compressed data in dest.

The compress() function is equivalent to compress2() with a level of Z_DEFAULT_COMPRESSION.

Return Value
On success, compress2() shall return Z_OK. Otherwise, compress2() shall return a value to indicate the error.

Errors
On error, compress2() shall return a value as described below:

Z_BUF_ERROR
The buffer dest was not large enough to hold the compressed data.

Z_MEM_ERROR
Insufficient memory.

Z_STREAM_ERROR
The level was not Z_DEFAULT_COMPRESSION, or was not between 0 and 9.
compressBound

Name
comprssBound — compute compressed data size

Synopsis
#include <zlib.h>
int compressBound(uLong sourceLen);

Description
The compressBound() function shall estimate the size of buffer required to compress
sourceLen bytes of data using the compress() or compress2() functions. If suc-
sequent, the value returned shall be an upper bound for the size of buffer required to
compress sourceLen bytes of data, using the parameters stored in stream, in a single
call to compress() or compress2().

Return Value
The compressBound() shall return a value representing the upper bound of an array to
allocate to hold the compressed data in a single call to compress() or compress2().
This function may return a conservative value that may be larger than sourceLen.

Errors
None defined.
**crc32**

**Name**
crc32 — compute CRC-32 Checksum

**Synopsis**
```c
#include <zlib.h>
uLong crc32(uLong crc, const Bytef * buf, uInt len);
```

**Description**
The crc32() function shall compute a running Cyclic Redundancy Check checksum, as defined in ITU-T V.42. On entry, `crc` is the previous value for the checksum, and `buf` shall point to an array of `len` bytes of data to be added to this checksum. The `crc32()` function shall return the new checksum.

If `buf` is NULL (or Z_NULL), `crc32()` shall return the initial checksum.

**Return Value**
The `crc32()` function shall return the new checksum value.

**Errors**
None defined.

**Application Usage (informative)**
The following code fragment demonstrates typical usage of the `crc32()` function:
```c
uLong crc = crc32(0L, Z_NULL, 0);
while (read_buffer(buffer, length) != EOF) {
    crc = crc32(crc, buffer, length);
}  
if (crc != original_crc) error();
```
**deflate**

**Name**

*deflate* — compress data

**Synopsis**

```c
#include <zlib.h>
int deflate(z_streamp stream, int flush);
```

**Description**

The *deflate()* function shall attempt to compress data until either the input buffer is empty or the output buffer is full. The *stream* references a *z_stream* structure. Before the first call to *deflate()* , this structure should have been initialized by a call to *deflateInit2()*.

**Note:** *deflateInit2()* is only in the binary standard; source level applications should initialize *stream* via a call to *deflateInit()* or *deflateInit2()*.

In addition, the *stream* input and output buffers should have been initialized as follows:

- **next_in**
  - should point to the data to be compressed.

- **avail_in**
  - should contain the number of bytes of data in the buffer referenced by *next_in*.

- **next_out**
  - should point to a buffer where compressed data may be placed.

- **avail_out**
  - should contain the size in bytes of the buffer referenced by *next_out*

The *deflate()* function shall perform one or both of the following actions:

1. Compress input data from *next_in* and update *next_in*, *avail_in* and *total_in* to reflect the data that has been compressed.

2. Fill the output buffer referenced by *next_out*, and update *next_out*, *avail_out* and *total_out* to reflect the compressed data that has been placed there. If *flush* is not *Z_NO_FLUSH*, and *avail_out* indicates that there is still space in output buffer, this action shall always occur (see below for further details).

The *deflate()* function shall return when either *avail_in* reaches zero (indicating that all the input data has been compressed), or *avail_out* reaches zero (indicating that the output buffer is full).

On success, the *deflate()* function shall set the *adler* field of the *stream* to the *adler32()* checksum of all the input data compressed so far (represented by *total_in*).

If the *deflate()* function shall attempt to determine the type of input data, and set field *data_type* in *stream* to *Z_ASCII* if the majority of the data bytes fall within the ASCII (ISO 646) printable character range. Otherwise, it shall set *data_type* to *Z_BINARY*. This data type is informational only, and does not affect the compression algorithm.
LSB Core - Generic 5.0

**Note:** Future versions of the LSB may remove this requirement, since it is based on an outdated character set that does not support Internationalization, and does not affect the algorithm. It is included for information only at this release. Applications should not depend on this field.

**Flush Operation**

The parameter `flush` determines when compressed bits are added to the output buffer in `next_out`. If `flush` is `Z_NO_FLUSH`, `deflate()` may return with some data pending output, and not yet added to the output buffer.

If `flush` is `Z_SYNC_FLUSH`, `deflate()` shall flush all pending output to `next_out` and align the output to a byte boundary. A synchronization point is generated in the output.

If `flush` is `Z_FULL_FLUSH`, all output shall be flushed, as for `Z_SYNC_FLUSH`, and the compression state shall be reset. A synchronization point is generated in the output.

**Rationale:** `Z_SYNC_FLUSH` is intended to ensure that the compressed data contains all the data compressed so far, and allows a decompressor to reconstruct all of the input data. `Z_FULL_FLUSH` allows decompression to restart from this point if the previous compressed data has been lost or damaged. Flushing is likely to degrade the performance of the compression system, and should only be used where necessary.

If `flush` is set to `Z_FINISH`, all pending input shall be processed and `deflate()` shall return with `Z_STREAM_END` if there is sufficient space in the output buffer at `next_out`, as indicated by `avail_out`. If `deflate()` is called with `flush` set to `Z_FINISH` and there is insufficient space to store the compressed data, and no other error has occurred during compression, `deflate()` shall return `Z_OK`, and the application should call `deflate()` again with `flush` unchanged, and having updated `next_out` and `avail_out`.

If all the compression is to be done in a single step, `deflate()` may be called with `flush` set to `Z_FINISH` immediately after the stream has been initialized if `avail_out` is set to at least the value returned by `deflateBound()`.

**Return Value**

On success, `deflate()` shall return `Z_OK`, unless `flush` was set to `Z_FINISH` and there was sufficient space in the output buffer to compress all of the input data. In this case, `deflate()` shall return `Z_STREAM_END`. On error, `deflate()` shall return a value to indicate the error.

**Note:** If `deflate()` returns `Z_OK` and has set `avail_out` to zero, the function should be called again with the same value for `flush`, and with updated `next_out` and `avail_out` until `deflate()` returns with `Z_OK` (or `Z_STREAM_END` if `flush` is set to `Z_FINISH`) and a non-zero `avail_out`.

**Errors**

On error, `deflate()` shall return a value as described below, and set the `msg` field of `stream` to point to a string describing the error:

**Z_BUF_ERROR**

No progress is possible; either `avail_in` or `avail_out` was zero.

**Z_MEM_ERROR**

Insufficient memory.

**Z_STREAM_ERROR**

The state (as represented in `stream`) is inconsistent, or `stream` was `NULL`. 

© 2015 Linux Foundation
deflateBound

Name
deflateBound — compute compressed data size

Synopsis
#include <zlib.h>
int deflateBound(z_stream *stream, uLong sourceLen);

Description
The deflateBound() function shall estimate the size of buffer required to compress sourceLen bytes of data. If successful, the value returned shall be an upper bound for the size of buffer required to compress sourceLen bytes of data, using the parameters stored in stream, in a single call to deflate() with flush set to Z_FINISH.

On entry, stream should have been initialized via a call to deflateInit__() or deflateInit2__().

Return Value
The deflateBound() shall return a value representing the upper bound of an array to allocate to hold the compressed data in a single call to deflate(). If the stream is not correctly initialized, or is NULL, then deflateBound() may return a conservative value that may be larger than sourceLen.

Errors
None defined.
deflateCopy

Name
deflateCopy — copy compression stream

Synopsis
#include <zlib.h>
int deflateCopy(z_streamp dest, z_streamp source);

Description
The deflateCopy() function shall copy the compression state information in source to the uninitialized z_stream structure referenced by dest.

On successful return, dest will be an exact copy of the stream referenced by source. The input and output buffer pointers in next_in and next_out will reference the same data.

Return Value
On success, deflateCopy() shall return Z_OK. Otherwise it shall return a value less than zero to indicate the error.

Errors
On error, deflateCopy() shall return a value as described below:

Z_STREAM_ERROR
  The state in source is inconsistent, or either source or dest was NULL.

Z_MEM_ERROR
  Insufficient memory available.

Application Usage (informative)
This function can be useful when several compression strategies will be tried, for example when there are several ways of pre-processing the input data with a filter. The streams that will be discarded should then be freed by calling deflateEnd(). Note that deflateCopy() duplicates the internal compression state which can be quite large, so this strategy may be slow and can consume lots of memory.
deflateEnd

Name
deflateEnd — free compression stream state

Synopsis

```c
#include <zlib.h>
int deflateEnd(z_streamp stream);
```

Description

The `deflateEnd()` function shall free all allocated state information referenced by `stream`. All pending output is discarded, and unprocessed input is ignored.

Return Value

On success, `deflateEnd()` shall return `Z_OK`, or `Z_DATA_ERROR` if there was pending output discarded or input unprocessed. Otherwise it shall return `Z_STREAM_ERROR` to indicate the error.

Errors

On error, `deflateEnd()` shall return `Z_STREAM_ERROR`. The following conditions shall be treated as an error:

- The state in `stream` is inconsistent or inappropriate.
- `stream` is `NULL`.

© 2015 Linux Foundation
deflateInit2_

Name
deflateInit2_ — initialize compression system

Synopsis

#include <zlib.h>

int deflateInit2_ (z_stream *strm, int level, int method, int windowBits, int memLevel, int strategy, char *version, int stream_size);

Description

The deflateInit2_() function shall initialize the compression system. On entry, strm shall refer to a user supplied z_stream object (a z_stream_s structure). The following fields shall be set on entry:

zalloc

a pointer to an alloc_func function, used to allocate state information. If this is NULL, a default allocation function will be used.

zfree

a pointer to a free_func function, used to free memory allocated by the zalloc function. If this is NULL a default free function will be used.

opaque

If alloc_func is not NULL, opaque is a user supplied pointer to data that will be passed to the alloc_func and free_func functions.

If the version requested is not compatible with the version implemented, or if the size of the z_stream_s structure provided in stream_size does not match the size in the library implementation, deflateInit2_() shall fail, and return Z_VERSION_ERROR.

The level supplied shall be a value between 0 and 9, or the value Z_DEFAULT_COMPRESSION. A level of 1 requests the highest speed, while a level of 9 requests the highest compression. A level of 0 indicates that no compression should be used, and the output shall be the same as the input.

The method selects the compression algorithm to use. LSB conforming implementation shall support the Z_DEFLATED method, and may support other implementation defined methods.

The windowBits parameter shall be a base 2 logarithm of the window size to use, and shall be a value between 8 and 15. A smaller value will use less memory, but will result in a poorer compression ratio, while a higher value will give better compression but utilize more memory.

The memLevel parameter specifies how much memory to use for the internal state. The value of memLevel shall be between 1 and MAX_MEM_LEVEL. Smaller values use less memory but are slower, while higher values use more memory to gain compression speed.

The strategy parameter selects the compression strategy to use:

Z_DEFAULT_STRATEGY

use the system default compression strategy. Z_DEFAULT_STRATEGY is particularly appropriate for text data.
Z_FILTERED

use a compression strategy tuned for data consisting largely of small values with a fairly random distribution. Z_FILTERED uses more Huffman encoding and less string matching than Z_DEFAULT_STRATEGY.

Z_HUFFMAN_ONLY

force Huffman encoding only, with no string match.

The deflateInit2() function is not in the source standard; it is only in the binary standard. Source applications should use the deflateInit2() macro.

Return Value

On success, the deflateInit2() function shall return Z_OK. Otherwise, deflateInit2() shall return a value as described below to indicate the error.

Errors

On error, deflateInit2() shall return one of the following error indicators:

Z_STREAM_ERROR

Invalid parameter.

Z_MEM_ERROR

Insufficient memory available.

Z_VERSION_ERROR

The version requested is not compatible with the library version, or the z_stream size differs from that used by the library.

In addition, the msg field of the strm may be set to an error message.
deflateInit_

Name
deflateInit_ — initialize compression system

Synopsis
#include <zlib.h>
int deflateInit_(z_stream *stream, int level, const char *version,
int stream_size);

Description
The deflateInit_() function shall initialize the compression system. On entry,
stream shall refer to a user supplied z_stream object (a z_stream_s structure). The
following fields shall be set on entry:

zalloc
    a pointer to an alloc_func function, used to allocate state information. If this is
NULL, a default allocation function will be used.

zfree
    a pointer to a free_func function, used to free memory allocated by the zalloc
function. If this is NULL a default free function will be used.

opaque
    If alloc_func is not NULL, opaque is a user supplied pointer to data that will be
passed to the alloc_func and free_func functions.

If the version requested is not compatible with the version implemented, or if the size
of the z_stream_s structure provided in stream_size does not match the size in the
library implementation, deflateInit_() shall fail, and return Z_VERSION_ERROR.

The level supplied shall be a value between 0 and 9, or the value Z_DEFAULT_COMPRESSION. A level of 1 requests the highest speed, while a level of 9 requests the
highest compression. A level of 0 indicates that no compression should be used, and
the output shall be the same as the input.

The deflateInit_() function is not in the source standard; it is only in the binary
standard. Source applications should use the deflateInit() macro.

The deflateInit_() function is equivalent to

    deflateInit2_(stream, level, Z_DEFLATED, MAX_WBITS,
MAX_MEM_LEVEL,
Return Value

On success, the deflateInit() function shall return Z_OK. Otherwise, deflateInit() shall return a value as described below to indicate the error.

Errors

On error, deflateInit() shall return one of the following error indicators:

Z_STREAM_ERROR
  Invalid parameter.

Z_MEM_ERROR
  Insufficient memory available.

Z_VERSION_ERROR
  The version requested is not compatible with the library version, or the z_stream size differs from that used by the library.

In addition, the msg field of the stream may be set to an error message.
deflateParams

Name
deflateParams — set compression parameters

Synopsis
#include <zlib.h>
int deflateParams(z_streamp stream, int level, int strategy);

Description
The deflateParams() function shall dynamically alter the compression parameters for the compression stream object stream. On entry, stream shall refer to a user supplied z_stream object (a z_stream_s structure), already initialized via a call to deflateInit() or deflateInit2().

The level supplied shall be a value between 0 and 9, or the value Z_DEFAULT_COMPRESSION. A level of 1 requests the highest speed, while a level of 9 requests the highest compression. A level of 0 indicates that no compression should be used, and the output shall be the same as the input. If the compression level is altered by deflateParams(), and some data has already been compressed with this stream (i.e. total_in is not zero), and the new level requires a different underlying compression method, then stream shall be flushed by a call to deflate().

The strategy parameter selects the compression strategy to use:

Z_DEFAULT_STRATEGY
use the system default compression strategy. Z_DEFAULT_STRATEGY is particularly appropriate for text data.

Z_FILTERED
use a compression strategy tuned for data consisting largely of small values with a fairly random distribution. Z_FILTERED uses more Huffman encoding and less string matching than Z_DEFAULT_STRATEGY.

Z_HUFFMAN_ONLY
force Huffman encoding only, with no string match.

Return Value
On success, the deflateParams() function shall return Z_OK. Otherwise, deflateParams() shall return a value as described below to indicate the error.

Errors
On error, deflateParams() shall return one of the following error indicators:

Z_STREAM_ERROR
Invalid parameter.

Z_MEM_ERROR
Insufficient memory available.

Z_BUF_ERROR
Insufficient space in stream to flush the current output.
In addition, the `msg` field of the `strm` may be set to an error message.

**Application Usage (Informative)**

Applications should ensure that the `stream` is flushed, e.g. by a call to `deflate(stream, Z_SYNC_FLUSH)` before calling `deflateParams()`, or ensure that there is sufficient space in `next_out` (as identified by `avail_out`) to ensure that all pending output and all uncompressed input can be flushed in a single call to `deflate()`.

**Rationale:** Although the `deflateParams()` function should flush pending output and compress all pending input, the result is unspecified if there is insufficient space in the output buffer. Applications should only call `deflateParams()` when the `stream` is effectively empty (flushed).

The `deflateParams()` can be used to switch between compression and straight copy of the input data, or to switch to a different kind of input data requiring a different strategy.

### deflateReset

**Name**

deflateReset — reset compression stream state

**Synopsis**

```c
#include <zlib.h>
int deflateReset(z_streamp stream);
```

**Description**

The `deflateReset()` function shall reset all state associated with `stream`. All pending output shall be discarded, and the counts of processed bytes (`total_in` and `total_out`) shall be reset to zero.

**Return Value**

On success, `deflateReset()` shall return `Z_OK`. Otherwise it shall return `Z_STREAM_ERROR` to indicate the error.

**Errors**

On error, `deflateReset()` shall return `Z_STREAM_ERROR`. The following conditions shall be treated as an error:

- The state in `stream` is inconsistent or inappropriate.
- `stream` is `NULL`.

---

© 2015 Linux Foundation
deflateSetDictionary

Name
deflateSetDictionary — initialize compression dictionary

Synopsis
#include <zlib.h>
int deflateSetDictionary(z_streamp stream, const Bytef * dictionary, uInt dictlen);

Description
The deflateSetDictionary() function shall initialize the compression dictionary associated with stream using the dictlen bytes referenced by dictionary.

The implementation may silently use a subset of the provided dictionary if the dictionary cannot fit in the current window associated with stream (see deflateInit2()).

The application should ensure that the dictionary is sorted such that the most commonly used strings occur at the end of the dictionary.

If the dictionary is successfully set, the Adler32 checksum of the entire provided dictionary shall be stored in the adler member of stream. This value may be used by the decompression system to select the correct dictionary. The compression and decompression systems must use the same dictionary.

stream shall reference an initialized compression stream, with total_in zero (i.e. no data has been compressed since the stream was initialized).

Return Value
On success, deflateSetDictionary() shall return Z_OK. Otherwise it shall return Z_STREAM_ERROR to indicate an error.

Errors
On error, deflateSetDictionary() shall return a value as described below:

Z_STREAM_ERROR

The state in stream is inconsistent, or stream was NULL.

Application Usage (informative)
The application should provide a dictionary consisting of strings that are likely to be encountered in the data to be compressed. The application should ensure that the dictionary is sorted such that the most commonly used strings occur at the end of the dictionary.

The use of a dictionary is optional; however if the data to be compressed is relatively short and has a predictable structure, the use of a dictionary can substantially improve the compression ratio.
get_crc_table

Name
get_crc_table — generate a table for crc calculations

Synopsis

#include <zlib.h>
const uLongf * get_crc_table(void);

Description

Generate tables for a byte-wise 32-bit CRC calculation based on the polynomial:
\[ x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + 1 \]

In a multi-threaded application, get_crc_table() should be called by one thread to initialize the tables before any other thread calls any libz function.

Return Value

The get_crc_table() function shall return a pointer to the first of a set of tables used internally to calculate CRC-32 values (see crc32()).

Errors

None defined.
gzclose

Name

gzclose — close a compressed file stream

Synopsis

#include <zlib.h>
int gzclose (gzFile file);

Description

The gzclose() function shall close the compressed file stream file. If file was open for writing, gzclose() shall first flush any pending output. Any state information allocated shall be freed.

Return Value

On success, gzclose() shall return Z_OK. Otherwise, gzclose() shall return an error value as described below.

Errors

On error, gzclose() may set the global variable errno to indicate the error. The gzclose() shall return a value other than Z_OK on error.

Z_STREAM_ERROR

file was NULL (or Z_NULL), or did not refer to an open compressed file stream.

Z_ERRNO

An error occurred in the underlying base libraries, and the application should check errno for further information.

Z_BUF_ERROR

no compression progress is possible during buffer flush (see deflate()).
**gzdopen**

**Name**

`gzdopen` — open a compressed file

**Synopsis**

```c
#include <zlib.h>
gzFile gzdopen ( int fd, const char *mode );
```

**Description**

The `gzdopen()` function shall attempt to associate the open file referenced by `fd` with a `gzFile` object. The `mode` argument is based on that of `fopen()`, but the `mode` parameter may also contain the following characters:

- **`digit`**
  
  set the compression level to `digit`. A low value (e.g. 1) means high speed, while a high value (e.g. 9) means high compression. A compression level of 0 (zero) means no compression. See `deflateInit2`() for further details.

- **`[fhR]`**
  
  set the compression strategy to `[fhR]`. The letter `f` corresponds to filtered data, the letter `h` corresponds to Huffman only compression, and the letter `R` corresponds to Run Length Encoding. See `deflateInit2()` for further details.

If `fd` refers to an uncompressed file, and `mode` refers to a read mode, `gzdopen()` shall attempt to open the file and return a `gzFile` object suitable for reading directly from the file without any decompression.

If `mode` is `NULL`, or if `mode` does not contain one of `r`, `w`, or `a`, `gzdopen()` shall return `Z_NULL`, and need not set any other error condition.

**Example**

```c
gzdopen(fileno(stdin), "r");
```

Attempt to associate the standard input with a `gzFile` object.

**Return Value**

On success, `gzdopen()` shall return a `gzFile` object. On failure, `gzdopen()` shall return `Z_NULL` and may set `errno` accordingly.

**Note:** At version 1.2.2, `zlib` does not set `errno` for several error conditions. Applications may not be able to determine the cause of an error.

**Errors**

On error, `gzdopen()` may set the global variable `errno` to indicate the error.
gzeof

Name
gzeof — check for end-of-file on a compressed file stream

Synopsis

#include <zlib.h>
int gzeof (gzFile file);

Description

The gzeof() function shall test the compressed file stream file for end of file.

Return Value

If file was open for reading and end of file has been reached, gzeof() shall return 1. Otherwise, gzeof() shall return 0.

Errors

None defined.

gzerror

Name
gzerror — decode an error on a compressed file stream

Synopsis

#include <zlib.h>
const char * gzerror (gzFile file, int * errnum);

Description

The gzerror() function shall return a string describing the last error to have occurred associated with the open compressed file stream referred to by file. It shall also set the location referenced by errnum to an integer value that further identifies the error.

Return Value

The gzerror() function shall return a string that describes the last error associated with the given file compressed file stream. This string shall have the format "%s: %s", with the name of the file, followed by a colon, a space, and the description of the error. If the compressed file stream was opened by a call to gzopen(), the format of the filename is unspecified.

Rationale: Although in all current implementations of libz file descriptors are named "<fd:%d>", the code suggests that this is for debugging purposes only, and may change in a future release.

It is unspecified if the string returned is determined by the setting of the LC_MESSAGES category in the current locale.

Errors

None defined.
gzflush

Name

gzflush — flush a compressed file stream

Synopsis

```c
#include <zlib.h>
int gzflush(gzFile file, int flush);
```

Description

The `gzflush()` function shall flush pending output to the compressed file stream identified by `file`, which must be open for writing.

Flush Operation

The parameter `flush` determines which compressed bits are added to the output file. If `flush` is `Z_NO_FLUSH`, `gzflush()` may return with some data pending output, and not yet written to the file.

If `flush` is `Z_SYNC_FLUSH`, `gzflush()` shall flush all pending output to `file` and align the output to a byte boundary. There may still be data pending compression that is not flushed.

If `flush` is `Z_FULL_FLUSH`, all output shall be flushed, as for `Z_SYNC_FLUSH`, and the compression state shall be reset. There may still be data pending compression that is not flushed.

Rationale: `Z_SYNC_FLUSH` is intended to ensure that the compressed data contains all the data compressed so far, and allows a decompressor to reconstruct all of the input data. `Z_FULL_FLUSH` allows decompression to restart from this point if the previous compressed data has been lost or damaged. Flushing is likely to degrade the performance of the compression system, and should only be used where necessary.

If `flush` is set to `Z_FINISH`, all pending uncompressed data shall be compressed and all output shall be flushed.

Return Value

On success, `gzflush()` shall return the value `Z_OK`. Otherwise `gzflush()` shall return a value to indicate the error, and may set the error number associated with the compressed file stream `file`.

Note: If `flush` is set to `Z_FINISH` and the flush operation is successful, `gzflush()` will return `Z_OK`, but the compressed file stream error value may be set to `Z_STREAM_END`.

Errors

On error, `gzflush()` shall return an error value, and may set the error number associated with the stream identified by `file` to indicate the error. Applications may use `gzerror()` to access this error value.

`Z_ERRNO`

An underlying base library function has indicated an error. The global variable `errno` may be examined for further information.

`Z_STREAM_ERROR`

The stream is invalid, is not open for writing, or is in an invalid state.
Z_BUF_ERROR

no compression progress is possible (see deflate()).

Z_MEM_ERROR

Insufficient memory available to compress.

gzgetc

Name
gzgetc — read a character from a compressed file

Synopsis

#include <zlib.h>
int gzgetc (gzFile file);

Description

The gzgetc() function shall read the next single character from the compressed file stream referenced by file, which shall have been opened in a read mode (see gzopen() and gzdopen()).

Return Value

On success, gzgetc() shall return the uncompressed character read, otherwise, on end of file or error, gzgetc() shall return -1.

Errors

On end of file or error, gzgetc() shall return -1. Further information can be found by calling gzerror() with a pointer to the compressed file stream.
gzgets

Name

gzgets — read a string from a compressed file

Synopsis

#include <zlib.h>
char * gzgets (gzFile file, char * buf, int len);

Description

The gzgets() function shall attempt to read data from the compressed file stream file, uncompressing it into buf until either len-1 bytes have been inserted into buf, or until a newline character has been uncompress into buf. A null byte shall be appended to the uncompressed data. The file shall have been opened in for reading (see gzopen() and gzdopen()).

Return Value

On success, gzgets() shall return a pointer to buf. Otherwise, gzgets() shall return Z_NULL. Applications may examine the cause using gzerror().

Errors

On error, gzgets() shall return Z_NULL. The following conditions shall always be treated as an error:

- file is NULL, or does not refer to a file open for reading;
- buf is NULL;
- len is less than or equal to zero.
gzopen

Name
gzopen — open a compressed file

Synopsis
#include <zlib.h>
gzFile gzopen (const char *path, const char *mode);

Description
The gzopen() function shall open the compressed file named by path. The mode argument is based on that of fopen(), but the mode parameter may also contain the following characters:

digit
set the compression level to digit. A low value (e.g. 1) means high speed, while a high value (e.g. 9) means high compression. A compression level of 0 (zero) means no compression. See deflateInit2() for further details.

[fhR]
set the compression strategy to [fhR]. The letter f corresponds to filtered data, the letter h corresponds to Huffman only compression, and the letter R corresponds to Run Length Encoding. See deflateInit2() for further details.

If path refers to an uncompressed file, and mode refers to a read mode, gzopen() shall attempt to open the file and return a gzFile object suitable for reading directly from the file without any decompression.

If path or mode is NULL, or if mode does not contain one of r, w, or a, gzopen() shall return Z_NULL, and need not set any other error condition.

The gzFile object is also referred to as a compressed file stream.

Example
gzopen("file.gz", "w6h");

Attempt to create a new compressed file, file.gz, at compression level 6 using Huffman only compression.

Return Value
On success, gzopen() shall return a gzFile object (also known as a compressed file stream). On failure, gzopen() shall return Z_NULL and may set errno accordingly.

Note: At version 1.2.2, zlib does not set errno for several error conditions. Applications may not be able to determine the cause of an error.

Errors
On error, gzopen() may set the global variable errno to indicate the error.
gzprintf

Name
gzprintf — format data and compress

Synopsis
#include <zlib.h>
int gzprintf (gzFile file, const char * fmt, ...);

Description
The gzprintf() function shall format data as for fprintf(), and write the resulting string to the compressed file stream file.

Return Value
The gzprintf() function shall return the number of uncompressed bytes actually written, or a value less than or equal to 0 in the event of an error.

Errors
If file is NULL, or refers to a compressed file stream that has not been opened for writing, gzprintf() shall return Z_STREAM_ERROR. Otherwise, errors are as for gzwrite().

gzputc

Name
gzputc — write character to a compressed file

Synopsis
#include <zlib.h>
int gzputc (gzFile file, int c);

Description
The gzputc() function shall write the single character c, converted from integer to unsigned character, to the compressed file referenced by file, which shall have been opened in a write mode (see gzopen() and gzdopen()).

Return Value
On success, gzputc() shall return the value written, otherwise gzputc() shall return -1.

Errors
On error, gzputc() shall return -1.
gzputs

Name

gzputs — string write to a compressed file

Synopsis

#include <zlib.h>
int gzputs (gzFile file, const char * s);

Description

The gzputs() function shall write the null terminated string s to the compressed file referenced by file, which shall have been opened in a write mode (see gzopen() and gzdopen()). The terminating null character shall not be written. The gzputs() function shall return the number of uncompressed bytes actually written.

Return Value

On success, gzputs() shall return the number of uncompressed bytes actually written to file. On error gzputs() shall return a value less than or equal to 0. Applications may examine the cause using gzerror().

Errors

On error, gzputs() shall set the error number associated with the stream identified by file to indicate the error. Applications should use gzerror() to access this error value. If file is NULL, gzputs() shall return Z_STREAM_ERR.

Z_ERRNO

An underlying base library function has indicated an error. The global variable errno may be examined for further information.

Z_STREAM_ERROR

The stream is invalid, is not open for writing, or is in an invalid state.

Z_BUF_ERROR

no compression progress is possible (see deflate()).

Z_MEM_ERROR

Insufficient memory available to compress.
gzread

Name
gzread — read from a compressed file

Synopsis
#include <zlib.h>
int gzread (gzFile file, voidp buf, unsigned int len);

Description
The gzread() function shall read data from the compressed file referenced by file, which shall have been opened in a read mode (see gzopen() and gzdopen()). The gzread() function shall read data from file, and uncompress it into buf. At most, len bytes of uncompressed data shall be copied to buf. If the file is not compressed, gzread() shall simply copy data from file to buf without alteration.

Return Value
On success, gzread() shall return the number of bytes decompressed into buf. If gzread() returns 0, either the end-of-file has been reached or an underlying read error has occurred. Applications should use gzerror() or gzeof() to determine which occurred. On other errors, gzread() shall return a value less than 0 and applications may examine the cause using gzerror().

Errors
On error, gzread() shall set the error number associated with the stream identified by file to indicate the error. Applications should use gzerror() to access this error value.

Z_ERRNO
An underlying base library function has indicated an error. The global variable errno may be examined for further information.

Z_STREAM_END
End of file has been reached on input.

Z_DATA_ERROR
A CRC error occurred when reading data; the file is corrupt.

Z_STREAM_ERROR
The stream is invalid, or is in an invalid state.

Z_NEED_DICT
A dictionary is needed (see inflateSetDictionary()).

Z_MEM_ERROR
Insufficient memory available to decompress.
gzrewind

Name

gzrewind — reset the file-position indicator on a compressed file stream

Synopsis

#include <zlib.h>
int gzrewind(gzFile file);

Description

The gzrewind() function shall set the starting position for the next read on compressed file stream file to the beginning of file. file must be open for reading. gzrewind() is equivalent to

(int)gzseek(file, 0L, SEEK_SET)

Return Value

On success, gzrewind() shall return 0. On error, gzrewind() shall return -1, and may set the error value for file accordingly.

Errors

On error, gzrewind() shall return -1, indicating that file is NULL, or does not represent an open compressed file stream, or represents a compressed file stream that is open for writing and is not currently at the beginning of file.
gzseek

Name
gzseek — reposition a file-position indicator in a compressed file stream

Synopsis
#include <zlib.h>

z_off_t gzseek(gzFile file, z_off_t offset, int whence);

Description
The gzseek() function shall set the file-position indicator for the compressed file stream file. The file-position indicator controls where the next read or write operation on the compressed file stream shall take place. The offset indicates a byte offset in the uncompressed data. The whence parameter may be one of:

SEEK_SET
   the offset is relative to the start of the uncompressed data.

SEEK_CUR
   the offset is relative to the current position in the uncompressed data.

Note: The value SEEK_END need not be supported.

If the file is open for writing, the new offset must be greater than or equal to the current offset. In this case, gzseek() shall compress a sequence of null bytes to fill the gap from the previous offset to the new offset.

Return Value
On success, gzseek() shall return the resulting offset in the file expressed as a byte position in the uncompressed data stream. On error, gzseek() shall return -1, and may set the error value for file accordingly.

Errors
On error, gzseek() shall return -1. The following conditions shall always result in an error:

• file is NULL
• file does not represent an open compressed file stream.
• file refers to a compressed file stream that is open for writing, and the newly computed offset is less than the current offset.
• The newly computed offset is less than zero.
• whence is not one of the supported values.

Application Usage (informative)
If file is open for reading, the implementation may still need to uncompress all of the data up to the new offset. As a result, gzseek() may be extremely slow in some circumstances.
gzsetparams

Name
gzsetparams — dynamically set compression parameters

Synopsis
#include <zlib.h>
int gzsetparams (gzFile file, int level, int strategy);

Description
The gzsetparams() function shall set the compression level and compression strategy
on the compressed file stream referenced by file. The compressed file stream shall
have been opened in a write mode. The level and strategy are as defined in
deflateInit2. If there is any data pending writing, it shall be flushed before the paramet-
ers are updated.

Return Value
On success, the gzsetparams() function shall return Z_OK.

Errors
On error, gzsetparams() shall return one of the following error indications:
Z_STREAM_ERROR
   Invalid parameter, or file not open for writing.
Z_BUF_ERROR
   An internal inconsistency was detected while flushing the previous buffer.
gztell

Name

gztell — find position on a compressed file stream

Synopsis

#include <zlib.h>

z_off_t gztell (gzFile file);

Description

The gztell() function shall return the starting position for the next read or write operation on compressed file stream file. This position represents the number of bytes from the beginning of file in the uncompressed data.

gztell() is equivalent to

gzseek(file, 0L, SEEK_CUR)

Return Value

gztell() shall return the current offset in the file expressed as a byte position in the uncompressed data stream. On error, gztell() shall return -1, and may set the error value for file accordingly.

Errors

On error, gztell() shall return -1, indicating that file is NULL, or does not represent an open compressed file stream.
gzwrite

Name

gzwrite — write to a compressed file

Synopsis

#include <zlib.h>
int gzwrite (gzFile file, void *buf, unsigned int len);

Description

The gzwrite() function shall write data to the compressed file referenced by file, which shall have been opened in a write mode (see gzopen() and gzdopen()). On entry, buf shall point to a buffer containing len bytes of uncompressed data. The gzwrite() function shall compress this data and write it to file. The gzwrite() function shall return the number of uncompressed bytes actually written.

Return Value

On success, gzwrite() shall return the number of uncompressed bytes actually written to file. On error gzwrite() shall return a value less than or equal to 0. Applications may examine the cause using gzerror().

Errors

On error, gzwrite() shall set the error number associated with the stream identified by file to indicate the error. Applications should use gzerror() to access this error value.

Z_ERRNO

An underlying base library function has indicated an error. The global variable errno may be examined for further information.

Z_STREAM_ERROR

The stream is invalid, is not open for writing, or is in an invalid state.

Z_BUF_ERROR

no compression progress is possible (see deflate()).

Z_MEM_ERROR

Insufficient memory available to compress.
infl{}ate

Name

infl{}ate — decompress data

Synopsis

#include <zlib.h>
int inflate(z_streamp stream, int flush);

Description

The inflate() function shall attempt to decompress data until either the input buffer is empty or the output buffer is full. The stream references a z_stream structure. Before the first call to inflate(), this structure should have been initialized by a call to inflateInit2().

Note: inflateInit2() is only in the binary standard; source level applications should initialize stream via a call to inflateInit() or inflateInit2().

In addition, the stream input and output buffers should have been initialized as follows:

next_in
    should point to the data to be decompressed.
avail_in
    should contain the number of bytes of data in the buffer referenced by next_in.
next_out
    should point to a buffer where decompressed data may be placed.
avail_out
    should contain the size in bytes of the buffer referenced by next_out.

The inflate() function shall perform one or both of the following actions:

1. Decompress input data from next_in and update next_in, avail_in and total_in to reflect the data that has been decompressed.
2. Fill the output buffer referenced by next_out, and update next_out, avail_out, and total_out to reflect the decompressed data that has been placed there. If flush is not Z_NO_FLUSH, and avail_out indicates that there is still space in output buffer, this action shall always occur (see below for further details).

The inflate() function shall return when either avail_in reaches zero (indicating that all the input data has been compressed), or avail_out reaches zero (indicating that the output buffer is full).

Flush Operation

The parameter flush determines when uncompressed bytes are added to the output buffer in next_out. If flush is Z_NO_FLUSH, inflate() may return with some data pending output, and not yet added to the output buffer.

If flush is Z_SYNC_FLUSH, inflate() shall flush all pending output to next_out, and update next_out and avail_out accordingly.
If \textit{flush} is set to 	exttt{Z\_BLOCK}, \texttt{inflate()} shall stop adding data to the output buffer if and when the next compressed block boundary is reached (see \texttt{RFC\_1951\_DEFLATE\_Compressed\_Data\_Format\_Specification}).

If \textit{flush} is set to 	exttt{Z\_FINISH}, all of the compressed input shall be decompressed and added to the output. If there is insufficient output space (i.e. the compressed input data uncompressed to more than \texttt{avail\_out} bytes), then \texttt{inflate()} shall fail and return \texttt{Z\_BUF\_ERROR}.

\section*{Return Value}

On success, \texttt{inflate()} shall return \texttt{Z\_OK} if decompression progress has been made, or \texttt{Z\_STREAM\_END} if all of the input data has been decompressed and there was sufficient space in the output buffer to store the uncompressed result. On error, \texttt{inflate()} shall return a value to indicate the error.

\begin{quote}
\textbf{Note}: If \texttt{inflate()} returns \texttt{Z\_OK} and has set \texttt{avail\_out} to zero, the function should be called again with the same value for \texttt{flush}, and with updated \texttt{next\_out} and \texttt{avail\_out} until \texttt{inflate()} returns with either \texttt{Z\_OK} or \texttt{Z\_STREAM\_END} and a non-zero \texttt{avail\_out}.
\end{quote}

On success, \texttt{inflate()} shall set the \texttt{adler} to the Adler-32 checksum of the output produced so far (i.e. \texttt{total\_out} bytes).

\section*{Errors}

On error, \texttt{inflate()} shall return a value as described below, and may set the \texttt{msg} field of \texttt{stream} to point to a string describing the error:

\texttt{Z\_BUF\_ERROR}

No progress is possible; either \texttt{avail\_in} or \texttt{avail\_out} was zero.

\texttt{Z\_MEM\_ERROR}

Insufficient memory.

\texttt{Z\_STREAM\_ERROR}

The state (as represented in \texttt{stream}) is inconsistent, or \texttt{stream} was NULL.

\texttt{Z\_NEED\_DICT}

A preset dictionary is required. The \texttt{adler} field shall be set to the Adler-32 checksum of the dictionary chosen by the compressor.
inflateEnd

Name
inflateEnd — free decompression stream state

Synopsis
#include <zlib.h>
int inflateEnd(z_streamp stream);

Description
The inflateEnd() function shall free all allocated state information referenced by stream. All pending output is discarded, and unprocessed input is ignored.

Return Value
On success, inflateEnd() shall return Z_OK. Otherwise it shall return Z_STREAM_ERROR to indicate the error.

Errors
On error, inflateEnd() shall return Z_STREAM_ERROR. The following conditions shall be treated as an error:
• The state in stream is inconsistent.
• stream is NULL.
• The zfree function pointer is NULL.
inflatableInit2_

Name
inflatableInit2_ — initialize decompression system

Synopsis
#include <zlib.h>
int inflatableInit2_ (z_streamp strm, int windowBits, char * version,
int stream_size);

Description
The inflatableInit2_( ) function shall initialize the decompression system. On entry, 
strm shall refer to a user supplied z_stream object (a z_stream_s structure). The fol-
lowing fields shall be set on entry:

zalloc
a pointer to an alloc_func function, used to allocate state information. If this is
NULL, a default allocation function will be used.

zfree
a pointer to a free_func function, used to free memory allocated by the zalloc
function. If this is NULL a default free function will be used.

opaque
If alloc_func is not NULL, opaque is a user supplied pointer to data that will be
passed to the alloc_func and free_func functions.

If the version requested is not compatible with the version implemented, or if the size
of the z_stream_s structure provided in stream_size does not match the size in the
library implementation, inflatableInit2_( ) shall fail, and return Z_VERSION_ERROR.

The windowBits parameter shall be a base 2 logarithm of the maximum window size
to use, and shall be a value between 8 and 15. If the input data was compressed with a
larger window size, subsequent attempts to decompress this data will fail with
Z_DATA_ERROR, rather than try to allocate a larger window.

The inflatableInit2_( ) function is not in the source standard; it is only in the binary
standard. Source applications should use the inflatableInit2() macro.

Return Value
On success, the inflatableInit2_( ) function shall return Z_OK. Otherwise, in-
flatableInit2_( ) shall return a value as described below to indicate the error.

Errors
On error, inflatableInit2_( ) shall return one of the following error indicators:

Z_STREAM_ERROR
Invalid parameter.

Z_MEM_ERROR
Insufficient memory available.

Z_VERSION_ERROR
The version requested is not compatible with the library version, or the \texttt{z\_stream} size differs from that used by the library.

In addition, the \texttt{msg} field of the \texttt{strm} may be set to an error message.
inflatableInit_

Name
inflatableInit_ — initialize decompression system

Synopsis
#include <zlib.h>
int inflatableInit_(z_stream * stream, const char * version, int stream_size);

Description
The inflatableInit_() function shall initialize the decompression system. On entry, stream shall refer to a user supplied z_stream object (a z_stream_s structure). The following fields shall be set on entry:

zalloc
a pointer to an alloc_func function, used to allocate state information. If this is NULL, a default allocation function will be used.

zfree
a pointer to a free_func function, used to free memory allocated by the zalloc function. If this is NULL a default free function will be used.

opaque
If alloc_func is not NULL, opaque is a user supplied pointer to data that will be passed to the alloc_func and free_func functions.

If the version requested is not compatible with the version implemented, or if the size of the z_stream_s structure provided in stream_size does not match the size in the library implementation, inflatableInit_() shall fail, and return Z_VERSION_ERROR.

The inflatableInit_() function is not in the source standard; it is only in the binary standard. Source applications should use the inflatableInit() macro.

The inflatableInit_() shall be equivalent to
inflatableInit2_(strm, MAX_WBITS, version, stream_size);

Return Value
On success, the inflatableInit_() function shall return Z_OK. Otherwise, inflatableInit_() shall return a value as described below to indicate the error.

Errors
On error, inflatableInit_() shall return one of the following error indicators:

Z_STREAM_ERROR
Invalid parameter.

Z_MEM_ERROR
Insufficient memory available.

Z_VERSION_ERROR
The version requested is not compatible with the library version, or the z_stream size differs from that used by the library.
In addition, the msg field of the strm may be set to an error message.

**inflateReset**

**Name**

inflateReset — reset decompression stream state

**Synopsis**

```c
#include <zlib.h>
int inflateReset(z_streamp stream);
```

**Description**

The `inflateReset()` function shall reset all state associated with `stream`. All pending output shall be discarded, and the counts of processed bytes (`total_in` and `total_out`) shall be reset to zero.

**Return Value**

On success, `inflateReset()` shall return Z_OK. Otherwise it shall return Z_STREAM_ERROR to indicate the error.

**Errors**

On error, `inflateReset()` shall return Z_STREAM_ERROR. The following conditions shall be treated as an error:

- The state in `stream` is inconsistent or inappropriate.
- `stream` is NULL.
inflateSetDictionary

Name
inflateSetDictionary — initialize decompression dictionary

Synopsis

#include <zlib.h>
int inflateSetDictionary(z_streamp stream, const Bytef * dictionary, uInt dictlen);

Description

The inflateSetDictionary() function shall initialize the decompression dictionary associated with stream using the dictlen bytes referenced by dictionary.

The inflateSetDictionary() function should be called immediately after a call to inflate() has failed with return value Z_NEED_DICT. The dictionary must have the same Adler-32 checksum as the dictionary used for the compression (see deflateSetDictionary()).

stream shall reference an initialized decompression stream, with total_in zero (i.e. no data has been decompressed since the stream was initialized).

Return Value

On success, inflateSetDictionary() shall return Z_OK. Otherwise it shall return a value as indicated below.

Errors

On error, inflateSetDictionary() shall return a value as described below:

Z_STREAM_ERROR

The state in stream is inconsistent, or stream was NULL.

Z_DATA_ERROR

The Adler-32 checksum of the supplied dictionary does not match that used for the compression.

Application Usage (informative)

The application should provide a dictionary consisting of strings {{ed note: do we really mean "strings"? Null terminated?}} that are likely to be encountered in the data to be compressed. The application should ensure that the dictionary is sorted such that the most commonly used strings occur at the end of the dictionary.

The use of a dictionary is optional; however if the data to be compressed is relatively short and has a predictable structure, the use of a dictionary can substantially improve the compression ratio.
inflateSync

Name

inflateSync — advance compression stream to next sync point

Synopsis

#include <zlib.h>
int inflateSync(z_stream *stream);

Description

The inflateSync() function shall advance through the compressed data in stream, skipping any invalid compressed data, until the next full flush point is reached, or all input is exhausted. See the description for deflate() with flush level Z_FULL_FLUSH. No output is placed in next_out.

Return Value

On success, inflateSync() shall return Z_OK, and update the next_in, avail_in, and total_in fields of stream to reflect the number of bytes of compressed data that have been skipped. Otherwise, inflateSync() shall return a value as described below to indicate the error.

Errors

On error, inflateSync() shall return a value as described below:

Z_STREAM_ERROR

The state (as represented in stream) is inconsistent, or stream was NULL.

Z_BUF_ERROR

There is no data available to skip over.

Z_DATA_ERROR

No sync point was found.
inflaterSyncPoint

Name
inflaterSyncPoint — test for synchronization point

Synopsis
#include <zlib.h>
int inflaterSyncPoint(z_stream *stream);

Description
The inflaterSyncPoint() function shall return a non-zero value if the compressed data stream referenced by stream is at a synchronization point.

Return Value
If the compressed data in stream is at a synchronization point (see deflate() with a flush level of Z_SYNC_FLUSH or Z_FULL_FLUSH), inflaterSyncPoint() shall return a non-zero value, other than Z_STREAM_ERROR. Otherwise, if the stream is valid, inflaterSyncPoint() shall return 0. If stream is invalid, or in an invalid state, inflaterSyncPoint() shall return Z_STREAM_ERROR to indicate the error.

Errors
On error, inflaterSyncPoint() shall return a value as described below:

Z_STREAM_ERROR
The state (as represented in stream) is inconsistent, or stream was NULL.
uncompress

Name
uncompress — uncompress data

Synopsis
#include <zlib.h>
int uncompress(Bytef * dest, uLongf * destLen, const Bytef * source, uLong sourceLen);

Description
The uncompress() function shall attempt to uncompress sourceLen bytes of data in the buffer source, placing the result in the buffer dest.

On entry, destLen should point to a value describing the size of the dest buffer. The application should ensure that this value is large enough to hold the entire uncompressed data.

Note: The LSB does not describe any mechanism by which a compressor can communicate the size required to the uncompressor.

On successful exit, the variable referenced by destLen shall be updated to hold the length of uncompressed data in dest.

Return Value
On success, uncompress() shall return Z_OK. Otherwise, uncompress() shall return a value to indicate the error.

Errors
On error, uncompress() shall return a value as described below:

Z_BUF_ERROR
The buffer dest was not large enough to hold the uncompressed data.

Z_MEM_ERROR
Insufficient memory.

Z_DATA_ERROR
The compressed data (referenced by source) was corrupted.
zError

**Name**

zError — translate error number to string

**Synopsis**

```c
#include <zlib.h>
const char * zError(int err);
```

**Description**

The zError() function shall return the string identifying the error associated with `err`. This allows for conversion from error code to string for functions such as compress() and uncompress(), that do not always set the string version of an error.

**Return Value**

The zError() function shall return a the string identifying the error associated with `err`, or NULL if `err` is not a valid error code.

It is unspecified if the string returned is determined by the setting of the LC_MESSAGES category in the current locale.

**Errors**

None defined.

zlibVersion

**Name**

zlibVersion — discover library version at run time

**Synopsis**

```c
#include <zlib.h>
const char * zlibVersion (void);
```

**Description**

The zlibVersion() function shall return the string identifying the interface version at the time the library was built.

Applications should compare the value returned from zlibVersion() with the macro constant ZLIB_VERSION for compatibility.

**Return Value**

The zlibVersion() function shall return a the string identifying the version of the library currently implemented.

**Errors**

None defined.

15.5 Interfaces for libncurses

Table 15-3 defines the library name and shared object name for the libncurses library
The parameters or return types of the following interfaces have had the const qualifier added as shown here, as compared to the specification in X/Open Curses, Issue 7.

```c
extern const char *keyname (int);
extern SCREEN *newterm (const char *, FILE *, FILE *);
extern const char *uncrlc (chtype);

extern int mvprintw (int, int, const char *, ...);
extern int mvwprintw (WINDOW *, int, int, const char *, ...);
extern int printw (const char *, ...);
extern int vwprintw (WINDOW *, const char *, va_list);
extern int wprintw (WINDOW *, const char *, ...);
extern int mvscanw (int, int, const char *, ...);
extern int mvwscanw (WINDOW *, int, int, const char *, ...);
extern int scanw (const char *, ...);
extern int vwscanw (WINDOW *, const char *, va_list);
extern int wscanw (WINDOW *, const char *, ...);
```

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

[LSB] This Specification
[X-CURSES] X/Open Curses, Issue 7

## 15.5.1 Curses

### 15.5.1.1 Interfaces for Curses

An LSB conforming implementation shall provide the generic functions for Curses specified in Table 15-4, with the full mandatory functionality as described in the referenced underlying specification.

### Table 15-4 libncurses - Curses Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>addch</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>addchstr</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>addnstr</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>addstr</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>attr_get</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>attr_off</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>attr_on</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>attrset</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>baudrate</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>beep</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>bkgd</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>bkgdset</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>border</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>box</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>can_change_color</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>cbreak</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>chgat</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>clear</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>clearok</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>clrtobot</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>clrtoeol</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>color_content</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>color_set</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>copywin</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>curs_set</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>def_prog_mode</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>def_shell_mode</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>del_curterm</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>delay_output</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>delch</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>deleteln</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>delscreen</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>delwin</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>derwin</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>doupdate</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>dupwin</td>
<td>[X-CURSES]</td>
</tr>
<tr>
<td>CURSES</td>
<td>CURSES</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>echo</td>
<td>echochar</td>
</tr>
<tr>
<td>erasechar</td>
<td>filter</td>
</tr>
<tr>
<td>getbkgd</td>
<td>getch</td>
</tr>
<tr>
<td>getwin</td>
<td>halfdelay</td>
</tr>
<tr>
<td>has_il</td>
<td>hline</td>
</tr>
<tr>
<td>immedok</td>
<td>inch</td>
</tr>
<tr>
<td>init_color</td>
<td>init_pair</td>
</tr>
<tr>
<td>insch</td>
<td>insdelln</td>
</tr>
<tr>
<td>instr</td>
<td>instr</td>
</tr>
<tr>
<td>is_wintouched</td>
<td>isendwin</td>
</tr>
<tr>
<td>killchar</td>
<td>leaveok</td>
</tr>
<tr>
<td>move</td>
<td>mvaddch</td>
</tr>
<tr>
<td>mvaddnstr</td>
<td>mvaddstr</td>
</tr>
<tr>
<td>mvdelch</td>
<td>mvderwin</td>
</tr>
<tr>
<td>mvgetstr</td>
<td>mvhline</td>
</tr>
<tr>
<td>mvinchstr</td>
<td>mvinstr</td>
</tr>
<tr>
<td>mvinstr</td>
<td>mvinstr</td>
</tr>
<tr>
<td>mvline</td>
<td>mvwaddch</td>
</tr>
<tr>
<td>mvwaddnstr</td>
<td>mvwaddstr</td>
</tr>
<tr>
<td>mvwgetch</td>
<td>mvwgetnstr</td>
</tr>
<tr>
<td>mvwin</td>
<td>mvwinch</td>
</tr>
<tr>
<td>mvwinnstr</td>
<td>mvwinsch</td>
</tr>
<tr>
<td>mvwinnstr</td>
<td>mvwprintw</td>
</tr>
<tr>
<td>napms</td>
<td>newpad</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>nl [X-CURSES]</td>
<td>nocbreak [X-CURSES]</td>
</tr>
<tr>
<td>nonl [X-CURSES]</td>
<td>nodelay [X-CURSES]</td>
</tr>
<tr>
<td>overlay [X-CURSES]</td>
<td>pechochar [X-CURSES]</td>
</tr>
<tr>
<td>nocbreak [X-CURSES]</td>
<td>put [X-CURSES]</td>
</tr>
<tr>
<td>nodelay [X-CURSES]</td>
<td>redrawwin [X-CURSES]</td>
</tr>
<tr>
<td>noecho [X-CURSES]</td>
<td>refresh [X-CURSES]</td>
</tr>
<tr>
<td>nonl [X-CURSES]</td>
<td>reset_prog_mode [X-CURSES]</td>
</tr>
<tr>
<td>noqiflush [X-CURSES]</td>
<td>reset_shell_mode [X-CURSES]</td>
</tr>
<tr>
<td>noraw [X-CURSES]</td>
<td>resetty [X-CURSES]</td>
</tr>
<tr>
<td>notimeout [X-CURSES]</td>
<td>ripoffline [LSB]</td>
</tr>
<tr>
<td>overlay [X-CURSES]</td>
<td>savetty [X-CURSES]</td>
</tr>
<tr>
<td>overwrite [X-CURSES]</td>
<td>scanw [LSB]</td>
</tr>
<tr>
<td>pair_content [X-CURSES]</td>
<td>set_curterm [X-CURSES]</td>
</tr>
<tr>
<td>pechochar [X-CURSES]</td>
<td>slk_attr_set [X-CURSES]</td>
</tr>
<tr>
<td>putp [X-CURSES]</td>
<td>slk_attroff [X-CURSES]</td>
</tr>
<tr>
<td>putwin [X-CURSES]</td>
<td>slk_attron [X-CURSES]</td>
</tr>
<tr>
<td>qiflush [X-CURSES]</td>
<td>slk_attrset [X-CURSES]</td>
</tr>
<tr>
<td>raw [X-CURSES]</td>
<td>slk_clear [X-CURSES]</td>
</tr>
<tr>
<td>redrawwin [X-CURSES]</td>
<td>slk_color [X-CURSES]</td>
</tr>
<tr>
<td>refresh [X-CURSES]</td>
<td>slkColorBrush [X-CURSES]</td>
</tr>
<tr>
<td>reset_prog_mode [X-CURSES]</td>
<td>slk_noutrefresh [X-CURSES]</td>
</tr>
<tr>
<td>reset_shell_mode [X-CURSES]</td>
<td></td>
</tr>
<tr>
<td>set_curterm [X-CURSES]</td>
<td>slk_refresh [X-CURSES]</td>
</tr>
<tr>
<td>set_term [X-CURSES]</td>
<td>slk_set [X-CURSES]</td>
</tr>
<tr>
<td>setscreegin [X-CURSES]</td>
<td>slk_touch [X-CURSES]</td>
</tr>
<tr>
<td>setupterm [X-CURSES]</td>
<td>std@end [X-CURSES]</td>
</tr>
<tr>
<td>ship_attrroff [X-CURSES]</td>
<td>standout [X-CURSES]</td>
</tr>
<tr>
<td>slk_attron [X-CURSES]</td>
<td>start_color [X-CURSES]</td>
</tr>
<tr>
<td>slk_attrset [X-CURSES]</td>
<td>subpad [X-CURSES]</td>
</tr>
<tr>
<td>slk_color [X-CURSES]</td>
<td>syncok [X-CURSES]</td>
</tr>
<tr>
<td>slk_init [X-CURSES]</td>
<td>termattrs [X-CURSES]</td>
</tr>
<tr>
<td>slk_label [X-CURSES]</td>
<td>termname [X-CURSES]</td>
</tr>
<tr>
<td>slk_noutrefresh [X-CURSES]</td>
<td></td>
</tr>
<tr>
<td>slk_refresh [X-CURSES]</td>
<td>tgetflag [X-CURSES]</td>
</tr>
<tr>
<td>slk_set [X-CURSES]</td>
<td>tgetnum [X-CURSES]</td>
</tr>
<tr>
<td>slk_touch [X-CURSES]</td>
<td>tgetstr [X-CURSES]</td>
</tr>
<tr>
<td>std@end [X-CURSES]</td>
<td>timeout [X-CURSES]</td>
</tr>
<tr>
<td>standout [X-CURSES]</td>
<td>touchline [X-CURSES]</td>
</tr>
<tr>
<td>subpad [X-CURSES]</td>
<td>touchwin [X-CURSES]</td>
</tr>
<tr>
<td>syncok [X-CURSES]</td>
<td>tparm [X-CURSES]</td>
</tr>
<tr>
<td>termattrs [X-CURSES]</td>
<td>typeahead [X-CURSES]</td>
</tr>
<tr>
<td>termname [X-CURSES]</td>
<td>unctrl [X-CURSES]</td>
</tr>
<tr>
<td>tgetflag [X-CURSES]</td>
<td>ungetch [X-CURSES]</td>
</tr>
<tr>
<td>tgetnum [X-CURSES]</td>
<td>vidattr [X-CURSES]</td>
</tr>
<tr>
<td>tgetstr [X-CURSES]</td>
<td>vidputs [X-CURSES]</td>
</tr>
<tr>
<td>tparm [X-CURSES]</td>
<td>use_env [X-CURSES]</td>
</tr>
<tr>
<td>typeahead [X-CURSES]</td>
<td>vw_printw [LSB]</td>
</tr>
<tr>
<td>unctrl [X-CURSES]</td>
<td>vw_scanw [LSB]</td>
</tr>
<tr>
<td>ungetch [X-CURSES]</td>
<td>vwprintw [X-CURSES]</td>
</tr>
<tr>
<td>use_env [X-CURSES]</td>
<td>vwscanw [LSB]</td>
</tr>
<tr>
<td>vw_printw [LSB]</td>
<td>waddch [X-CURSES]</td>
</tr>
<tr>
<td>vw_scanw [LSB]</td>
<td>waddchstr [X-CURSES]</td>
</tr>
<tr>
<td>vwprintw [X-CURSES]</td>
<td>waddnstr [X-CURSES]</td>
</tr>
<tr>
<td>waddnstr [X-CURSES]</td>
<td>wattr_get [X-CURSES]</td>
</tr>
<tr>
<td>wattr_off [X-CURSES]</td>
<td>wattr_on [X-CURSES]</td>
</tr>
<tr>
<td>wattr_off [X-CURSES]</td>
<td>wattron [X-CURSES]</td>
</tr>
<tr>
<td>wattr_set [X-CURSES]</td>
<td>wattron [X-CURSES]</td>
</tr>
<tr>
<td>wattron [X-CURSES]</td>
<td>wborder [X-CURSES]</td>
</tr>
<tr>
<td>wattron [X-CURSES]</td>
<td>wborder [X-CURSES]</td>
</tr>
<tr>
<td>wborder [X-CURSES]</td>
<td>wchgdset [X-CURSES]</td>
</tr>
<tr>
<td>wchgdset [X-CURSES]</td>
<td>wchgdset [X-CURSES]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic deprecated functions for Curses specified in Table 15-5, 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 15-5 libncurses - Curses Deprecated Function Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Interface</th>
<th>Interface</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>tgoto [X-CURSES]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Curses specified in Table 15-6, with the full mandatory functionality as described in the referenced underlying specification.

Table 15-6 libncurses - Curses Data Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Interface</th>
<th>Interface</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>stdscr [X-CURSES]</td>
<td>tytype [X-CURSES]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

15.6 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.

### 15.6.1 curses.h

```
#define getattrs(win)   ((win)?(win)->_attrs:A_NORMAL)
#define ERR (-1)
#define OK (0)
#define ACS_RARROW      (acs_map[+'+'])
#define ACS_LARROW      (acs_map[+'-'])
#define ACS_UARROW      (acs_map[+'-'])
#define ACS_DARROW      (acs_map[+'-'])
#define ACS_BLOCK       (acs_map[+'0'])
#define ACS_CKBOARD     (acs_map[+'a'])
#define ACS_DEGREE      (acs_map[+'f'])
#define ACS_PLMINUS     (acs_map[+'g'])
#define ACS_BOARD       (acs_map[+'h'])
#define ACS_LANTERN     (acs_map[+'i'])
#define ACS_LRCORNER    (acs_map[+'j'])
#define ACS_URCORNER    (acs_map[+'k'])
#define ACS_ULCORNER    (acs_map[+'l'])
#define ACS_LLCORNER    (acs_map[+'m'])
#define ACS_PLUS        (acs_map[+'n'])
#define ACS_S1  (acs_map[+'o'])
#define ACS_HLINE       (acs_map[+'q'])
#define ACS_S9  (acs_map[+'s'])
#defineACS_LTEE        (acs_map[+'t'])
#define ACS_BTEE        (acs_map[+'u'])
#define ACS_TTEE        (acs_map[+'v'])
#define ACS_VLINE       (acs_map[+'w'])
#define ACS_DIAMOND     (acs_map[+'x'])
#define ACS_BULLET      (acs_map[+'y'])
#define vid_attr(a,pair,opts)   vidattr(a)
#define getmaxyx(win,y,x)       \(y=(win)?((win)->_maxy+1):ERR,x=(win)?((win)\->_maxx+1):ERR\)
#define getbegyx(win,y,x)       \(y=(win)?((win)->_begy:ERR,x=(win)?((win)->_begx:ERR\)
#define getparyx(win,y,x)       \(y=(win)?((win)->_pary:ERR,x=(win)?((win)->_parx:ERR\)
```

```
#define __NCURSES_H     1
#define NCURSES_EXPORT(type)    type
#define NCURSES_EXPORT_VAR(type)        type
#define WA_ALTCHARSET   A_ALTCHARSET
#define WA_ATTRIBUTES   A_ATTRIBUTES
#define WA_BLINK        A_BLINK
#define WA_BOLD  A_BOLD
```
```c
#define WA_DIM A_DIM
#define WA_HORIZONTAL A_HORIZONTAL
#define WA_INVIS A_INVIS
#define WA_LEFT A_LEFT
#define WA_LOW A_LOW
#define WA_NORMAL A_NORMAL
#define WA_PROTECT A_PROTECT
#define WA_REVERSE A_REVERSE
#define WA_RIGHT A_RIGHT
#define WA_STANDOUT A_STANDOUT
#define WA_TOP A_TOP
#define WA_UNDERLINE A_UNDERLINE
#define A_REVERSE NCURSES_BITS(1UL,10)
#define COLOR_BLACK 0
#define COLOR_RED 1
#define COLOR_GREEN 2
#define COLOR_YELLOW 3
#define COLOR_BLUE 4
#define COLOR_MAGENTA 5
#define COLOR_CYAN 6
#define COLOR_WHITE 7
#define _SUBWIN 0x01
#define _ENDLINE 0x02
#define _FULLWIN 0x04
#define _SCROLLWIN 0x08
#define _ISPAD 0x10
#define _HASMOVED 0x20

typedef unsigned char bool;
typedef unsigned long int chtype;
typedef struct screen SCREEN;
typedef struct _win_st WINDOW;
typedef chtype attr_t;
typedef struct { attr_t attr;
               wchar_t chars[5];
} cchar_t;
struct pdat { short _pad_y;
              short _pad_x;
              short _pad_top;
              short _pad_left;
              short _pad_bottom;
              short _pad_right;
};

struct _win_st { short _cury; /* current cursor position */
                 short _curx;
                 short _maxy; /* maximums of x and y, NOT
window size */
                 short _maxx;
                 short _begy; /* screen coords of upper-left-
hand corner */
                 short _begx;
                 short _flags; /* window state flags */
                 attr_t _attrs; /* current attribute for non-
space character */
                 chtype _bkgd; /* current background
char/attribute pair */
                 bool _notimeout; /* no time out on function-key
entry? */
```
bool _clear;              /* consider all data in the
window invalid? */
bool _leaveok;              /* OK to not reset cursor on
exit? */
bool _scroll;               /* OK to scroll this window? */
bool _idlok;                /* OK to use insert/delete line?
 */
bool _idcok;                /* OK to use insert/delete char?
 */
bool _immed;                /* window in immed mode? (not yet
used) */
bool _sync;                 /* window in sync mode? */
    /* process function keys into
KEY_ symbols? */
int _delay;                 /* 0 = nodelay, <0 = blocking, >0
delay */
= delay */
    struct ldat *_line;
    /* the actual line data */
short _regtop;              /* top line of scrolling region
 */
short _regbottom;           /* bottom line of scrolling
region */
int _parx;                  /* x coordinate of this window in
parent */
int _pary;                  /* y coordinate of this window in
parent */
WINDOW * _parent;            /* pointer to parent if a sub-
window */
    struct pdat _pad;
    /* real begy is _begy + _yoffset
 */
short _yoffset;             /* real begy is _begy + _yoffset
 */
cchar_t _bkgrnd;            /* current background
char/attribute pair */
};

#define KEY_F(n)        (KEY_F0+(n))
#define KEY_CODE_YES    0400
#define KEY_BREAK       0401
#define KEY_MIN 0401
#define KEY_DOWN        0402
#define KEY_UP  0403
#define KEY_LEFT        0404
#define KEY_RIGHT       0405
#define KEY_HOME        0406
#define KEY_BACKSPACE   0407
#define KEY_F0  0410
#define KEY_DL  0510
#define KEY_IL 0511
#define KEY_DC  0512
#define KEY_IC  0513
#define KEY_EIC 0514
#define KEY_CLEAR       0515
#define KEY_EOL 0517
#define KEY_SF  0520
#define KEY_SR 0521
#define KEY_NPAGE      0522
#define KEY_PPAGE 0523
#define KEY_STAB 0524
#define KEY_CTAB 0525
#define KEY_CATAB 0526
#define KEY_ENTER 0527
#define KEY_EOL 0528
#define KEY_SRESET      0530
#define KEY_RESET       0531
#define KEY_PRINT       0532
#define KEY_LL 0533
#define KEY_A1 0534
#define KEY_A3  0535
#define KEY_B2  0536
#define KEY_C1  0537
#define KEY_C3  0540
#define KEY_BTAB  0541
#define KEY_BEG  0542
#define KEY_CANCEL  0543
#define KEY_CLOSE  0544
#define KEY_COMMAND  0547
#define KEY_COPY  0546
#define KEY_CREATE  0547
#define KEY_END  0550
#define KEY_EXIT  0551
#define KEY_FIND  0552
#define KEY_HELP  0553
#define KEY_MARK  0554
#define KEY_MESSAGE  0555
#define KEY_MOVE  0556
#define KEY_NEXT  0557
#define KEY_OPEN  0558
#define KEY_OPTIONS  0561
#define KEY_PREVIOUS  0562
#define KEY_REDO  0563
#define KEY_REFERENCE  0564
#define KEY_REFRESH  0565
#define KEY_REPLACE  0566
#define KEY_RESTART  0567
#define KEY_RESUME  0570
#define KEY_SAVE  0571
#define KEY_SBEG  0572
#define KEY_SCANCEL  0573
#define KEY_SCOMMAND  0574
#define KEY_SDC  0577
#define KEY_SDLC  0600
#define KEY_SELECT  0601
#define KEY_SEND  0602
#define KEY_SEOL  0603
#define KEY_SEXIT  0604
#define KEY_SFIND  0605
#define KEY_SHELP  0606
#define KEY_SHOME  0607
#define KEY_SIC  0610
#define KEY_SLEFT  0611
#define KEY_SMESAGE  0612
#define KEY_SMOVE  0613
#define KEY_SNEXT  0614
#define KEY_SOPTIONS  0615
#define KEY_SPREVIOUS  0616
#define KEY_SPRINT  0617
#define KEY_SREDO  0620
#define KEY_SREPLACE  0621
#define KEY_SRIGHT  0622
#define KEY_SRSUME  0623
#define KEY_SSAVE  0624
#define KEY_SSUSPEND  0625
#define KEY_SUNDO  0626
#define KEY_SUSPEND  0627
#define KEY_UNDO  0630
#define KEY_MOUSE  0631
#define KEY_RESIZE  0632
#define KEY_MAX  0777

#define PAIR_NUMBER(a) (((a)&A_COLOR)>>8)
#define NCURSES_BITS(mask,shift) ((mask)<<((shift)+8))
```c
#define A_CHARTEXT      (NCURSES_BITS(1UL,0)-1UL)
#define A_NORMAL        0L
#define NCURSES_ATTR_SHIFT      8
#define A_COLOR NCURSES_BITS(((1UL)<<8)-1UL,0)
#define A_BLINK NCURSES_BITS(1UL,11)
#define A_DIM   NCURSES_BITS(1UL,12)
#define A_BOLD  NCURSES_BITS(1UL,13)
#define A_ALTCHARSET    NCURSES_BITS(1UL,14)
#define A_INVIS NCURSES_BITS(1UL,15)
#define A_PROTECT      NCURSES_BITS(1UL,16)
#define A_HORIZONTAL   NCURSES_BITS(1UL,17)
#define A_LEFT    NCURSES_BITS(1UL,18)
#define A_LOW     NCURSES_BITS(1UL,19)
#define A_RIGHT   NCURSES_BITS(1UL,20)
#define A_TOP     NCURSES_BITS(1UL,21)
#define A_VERTICAL  NCURSES_BITS(1UL,22)
#define A_STANDOUT   NCURSES_BITS(1UL,8)
#define A_UNDERLINE  NCURSES_BITS(1UL,9)
#define COLOR_PAIR(n)   NCURSES_BITS(n,0)
#define A_ATTRIBUTES   NCURSES_BITS(~(1UL-1UL),0)

extern int COLORS;
extern int COLOR_PAIRS;
extern int COLS;
extern int LINES;
extern chtype acs_map[];
extern int addch(const chtype);
extern int addchnstr(const chtype *, int);
extern int addchnstrs(const char *, int);
extern int addnstr(const char *, int);
extern int addstr(const char *);
extern int attr_get(attr_t *, short *, void *);
extern int attr_off(attr_t, void *);
extern int attr_on(attr_t, void *);
extern int attr_set(attr_t, short, void *);
extern int attroff(int);
extern int attron(int);
extern int attrset(int);
extern int baudrate(void);
extern int beep(void);
extern int bkgd(chtype);
extern void bkgdset(chtype);
extern int border(chtype, chtype, chtype, chtype, chtype, chtype, chtype, chtype);
extern int box(WINDOW *, chtype, chtype);
extern bool can_change_color(void);
extern int cbreak(void);
extern int chgat(int, attr_t, short, const void *);
extern int clear(void);
extern int clearok(WINDOW *, bool);
extern int clrtoeol(void);
extern int clrtobot(void);
extern int color_content(short, short *, short *, short *);
extern int color_set(short, void *);
extern int copywin(const short, short *, short *, short *, short *
 extern int copywin(const char *, int, int, int, int,
 extern int curs_set(int);
extern WINDOW *curscr;
extern int def_prog_mode(void);
extern int def_shell_mode(void);
extern int delay_output(int);
extern int delch(void);
extern int deletein(void);
extern void delscreen(SCREEN *);
```
extern int delwin(WINDOW *
); extern int derwin(WINDOW *, int, int, int, int, int
); extern int doupdate(void
); extern int dupwin(WINDOW *
); extern int echo(void
); extern int echochar(const chtype
); extern int endwin(void
); extern int erase(void
); extern char erasechar(void
); extern void filter(void
); extern int flash(void
); extern int flushinp(void
); extern chtype getbkgd(WINDOW *
); extern int getch(void
); extern int getnstr(char *, int
); extern int getstr(char *
); extern WINDOW *getwin(FILE *
); extern int halfdelay(int
); extern bool has_colors(void
); extern bool has_ic(void
); extern bool has_il(void
); extern int hline(chtype, int
); extern void idcok(WINDOW *, bool
); extern int idlok(WINDOW *, bool
); extern void immedok(WINDOW *, bool
); extern chtype inch(void
); extern int inchnstr(chtype *, int
); extern int inchstr(chtype *
); extern int init_color(short, short, short, short
); extern int init_pair(short, short, short
); extern WINDOW *initscr(void
); extern int innstr(char *, int
); extern int insch(chtype
); extern int insdelln(int
); extern int insnstr(const char *, int
); extern int insstr(const char *
); extern int instr(char *
); extern int intrflush(WINDOW *, bool
); extern bool is_linetouched(WINDOW *, int
); extern bool is_wintouched(WINDOW *
); extern bool isendwin(void
); extern const char *keyname(int
); extern int keypad(WINDOW *, bool
); extern char killchar(void
); extern int leaveok(WINDOW *, bool
); extern char *longname(void
); extern int move(int, int
); extern int mvaddch(int, int, const chtype *
); extern int mvaddchstr(int, int, const chtype *
); extern int mvaddnstr(int, int, const char *, int
); extern int mvaddstr(int, int, const char *
); extern int mvchgat(int, int, int, attr_t, short, const void *
); extern int mvcur(int, int, int
); extern int mvdelch(int
); extern int mvderwin(WINDOW *, int, int
); extern int mvgetch(int, int
); extern int mvgetnstr(int, int, char *, int
); extern int mvgetstr(int, int, char *
); extern int mvhline(int, int, chtype, int
); extern chtype mvinch(int, int
); extern int mvinnstr(int, int, chtype *, int
); extern int mvinchnstr(int, int, chtype *
); extern int mvinchnstr(int, int, char *, int
); extern int mvinsstr(int, int, char *, int
);
extern int mvinsch(int, int, chtype);
extern int mwinsnstr(int, int, const char *, int);
extern int mwinsnstr(int, int, const char *, int);
extern int mvinstr(int, int, char *);
extern int mvprintw(int, int, const char *, ...);
extern int mvscanw(int, int, const char *, ...);
extern int mvvline(int, int, chtype, int);
extern int mwaddch(WINDOW *, int, int, const chtype);
extern int mwaddchnstr(WINDOW *, int, int, const chtype *, int);
extern int mwaddchstr(WINDOW *, int, int, const chtype *);
extern int mwaddnstr(WINDOW *, int, int, const char *, int);
extern int mwaddstr(WINDOW *, int, int, const char *);
extern int mwchgat(WINDOW *, int, int, int, attr_t, short, const void *);
extern int mwdelch(WINDOW *, int, int);
extern int mwgetch(WINDOW *, int, int);
extern int mwgetnstr(WINDOW *, int, int, char *, int);
extern int mwgetline(WINDOW *, int, int, char *, int);
extern int mwprintw(WINDOW *, int, int, int, int, int, int, int);
chtype mwwinch(WINDOW *, int, int);
extern int mwwinchstr(WINDOW *, int, int, chtype *);
extern int mwwinsnstr(WINDOW *, int, int, const char *, int);
extern int mwwinsstr(WINDOW *, int, int, const char *);
extern int mwprintw(WINDOW *, int, int, const char *, ...);
extern int mwscanw(WINDOW *, int, int, const char *, ...);
extern int mwvline(WINDOW *, int, int, chtype, int);
extern int napms(int);
extern WINDOW *newpad(int, int);
extern WINDOW *newscr;
extern SCREEN *newterm(const char *, FILE *, FILE *);
extern WINDOW *newwin(int, int, int, int);
extern int nl(void);
extern int nocbreak(void);
extern int nodelay(WINDOW *, bool);
extern int noecho(void);
extern int nonl(void);
extern void noqiflush(void);
extern int noraw(void);
extern int notimeout(WINDOW *, bool);
extern int overlay(const WINDOW *, WINDOW *);
extern int overwrite(const WINDOW *, WINDOW *);
extern int pair_content(short, short *, short *);
extern int pechochar(WINDOW *, chtype);
extern int pnoutrefresh(WINDOW *, int, int, int, int, int);
extern int prefresh(WINDOW *, int, int, int, int, int, int);
extern int printw(const char *, ...);
extern int putwin(WINDOW *, FILE *);
extern void qiflush(void);
extern int raw(void);
extern int redrawwin(WINDOW *);
extern int refresh(void);
extern int reset_prog_mode(void);
extern int reset_shell_mode(void);
extern int resetty(void);
extern int ripoffline(int, int (*)(WINDOW *, int));
extern int savetty(void);
extern int scanw(const char *, ...);
extern int scanw(const char *, ...);
extern int scr_dumps(const char *);
extern int scr_init(const char *);
extern int scr_restore(const char *);
extern int scr_set(const char *);
extern int wdelch(WINDOW *);
extern int wdeletein(WINDOW *);
extern int wechochar(WINDOW *, const chtype);
extern int werase(WINDOW *);
extern int wgetch(WINDOW *);
extern int wgetnstr(WINDOW *, char *, int);
extern int wgetstr(WINDOW *, char *);
extern int whline(WINDOW *, chtype, int);
extern chtype winch(WINDOW *);
extern int winchnstr(WINDOW *, chtype *, int);
extern int winchstr(WINDOW *, chtype *);
extern int winnstr(WINDOW *, char *, int);
extern int winsdelln(WINDOW *, int);
extern int winsinsertln(WINDOW *);
extern int winsnstr(WINDOW *, const char *, int);
extern int winsstr(WINDOW *, const char *);
extern int winstr(WINDOW *, char *);
extern int wmove(WINDOW *, int, int);
extern int wnoutrefresh(WINDOW *);
extern int wprintw(WINDOW *, const char *, ...);
extern int wrefresh(WINDOW *);
extern int wscrl(WINDOW *, int);
extern int wsetscrreg(WINDOW *, int, int);
extern int wstandend(WINDOW *);
extern int wstandout(WINDOW *);
extern void wsyncdown(WINDOW *);
extern void wsyncup(WINDOW *);
extern void wtimeout(WINDOW *, int);
extern int wtouchln(WINDOW *, int, int, int);
extern int wvline(WINDOW *, chtype, int);

15.6.2 term.h

extern TERMINAL *cur_term;
extern int del_curterm(TERMINAL *);
extern int putp(const char *);
extern int restartterm(char *, int, int *);
extern TERMINAL *set_curterm(TERMINAL *);
extern int setupterm(char *, int, int *);
extern int tgetent(char *, const char *);
extern int tgetflag(char *);
extern int tgetnum(char *);
extern char *tgetstr(char *, char **);
extern char *tgoto(const char *, int);
extern int tigetflag(const char *);
extern int tigetnum(const char *);
extern char *tigetstr(const char *);
extern char *tparm(const char *, ...);
extern int tputs(const char *, int, int (*)(int));
extern char ttytype[];

15.7 Interface Definitions for libncurses

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

Other interfaces listed in Section 15.5 shall behave as described in the referenced base document.
inchnstr

Name
ininchnstr — obtain a string of characters and their attributes from a curses window

Synopsis

```c
#include <curses.h>
int inchnstr(chtype * chstr, int n);
```

Description

The interface inchnstr() shall behave as specified in X/Open Curses, Issue 7, except that inchnstr() shall return the number of characters that were read.

inchstr

Name
ininchstr — obtain a string of characters and their attributes from a curses window

Synopsis

```c
#include <curses.h>
int inchstr(chtype * chstr);
```

Description

The interface inchstr() shall behave as specified in X/Open Curses, Issue 7, except that inchstr() shall return the number of characters that were read.

instr

Name
inistr — obtain a string of characters from a curses window

Synopsis

```c
#include <curses.h>
int instr(char * str);
```

Description

The interface instr() shall behave as specified in X/Open Curses, Issue 7, except that instr() shall return the number of characters that were read.
mvcur

Name
mvcur — send cursor movement commands to terminal

Synopsis
#include <curses.h>
int mvcur(int oldrow, int oldcol, int newrow, int newcol);

Description
The interface mvcur() shall behave as described in X/Open Curses, Issue 7, except that
if (newrow, newcol) is not a valid address for the terminal in use, the results of the mv-
cur() function are unspecified.

mvinchnstr

Name
mvinchnstr — obtain a string of characters and their attributes from a curses window

Synopsis
#include <curses.h>
int mvinchnstr(int y, int x, chtype *chstr, int n);

Description
The interface mvinchnstr() shall behave as specified in X/Open Curses, Issue 7, except that
mvinchnstr() shall return the number of characters that were read.

mvinchstr

Name
mvinchstr — obtain a string of characters and their attributes from a curses window

Synopsis
#include <curses.h>
int mvinchstr(int y, int x, chtype *chstr);

Description
The interface mvinchstr() shall behave as specified in X/Open Curses, Issue 7, except
that mvinchstr() shall return the number of characters that were read.
mvinstr

Name
mvinstr — obtain a string of characters from a curses window

Synopsis
#include <curses.h>
int mvinstr(int y, int x, char * str);

Description
The interface mvinstr() shall behave as specified in X/Open Curses, Issue 7, except that mvinstr() shall return the number of characters that were read.

mvscanw

Name
mvscanw — convert formatted input from a curses window

Synopsis
#include <curses.h>
int mvscanw(int y, int x, const char * fmt, ...);

Description
The scanw family of functions shall behave as described in X/Open Curses, Issue 7, except as noted below.

Differences
This function returns ERR on failure. On success it returns the number of successfully matched and assigned input items. This differs from X/Open Curses, Issue 7, which indicates this function returns OK on success.

mvwinchnstr

Name
mvwinchnstr — obtain a string of characters and their attributes from a curses window

Synopsis
#include <curses.h>
int mvwinchnstr(WINDOW * win, int y, int x, chtype * chstr, int n);

Description
The interface mvwinchnstr() shall behave as specified in X/Open Curses, Issue 7, except that mvwinchnstr() shall return the number of characters that were read.
**mvwinchstr**

**Name**

`mvwinchstr` — obtain a string of characters and their attributes from a curses window

**Synopsis**

```c
#include <curses.h>
int mvwinchstr(WINDOW * win, int y, int x, chtype * chstr);
```

**Description**

The interface `mvwinchstr()` shall behave as specified in [X/Open Curses, Issue 7](https://www.opengroup.org/onlinepubs/007904975/xsh/curses.html), except that `mvwinchstr()` shall return the number of characters that were read.

**mvwinstr**

**Name**

`mvwinstr` — obtain a string of characters from a curses window

**Synopsis**

```c
#include <curses.h>
int mvwinstr(WINDOW * win, int y, int x, char * str);
```

**Description**

The interface `mvwinstr()` shall behave as specified in [X/Open Curses, Issue 7](https://www.opengroup.org/onlinepubs/007904975/xsh/curses.html), except that `mvwinstr()` shall return the number of characters that were read.

**mvwscanw**

**Name**

`mvwscanw` — convert formatted input from a curses window

**Synopsis**

```c
#include <curses.h>
int mvwscanw(WINDOW * win, int y, int x, const char * fmt, ...);
```

**Description**

The `scanw` family of functions shall behave as described in [X/Open Curses, Issue 7](https://www.opengroup.org/onlinepubs/007904975/xsh/curses.html), except as noted below.

**Differences**

This function returns `ERR` on failure. On success it returns the number of successfully matched and assigned input items. This differs from [X/Open Curses, Issue 7](https://www.opengroup.org/onlinepubs/007904975/xsh/curses.html), which indicates this function returns `OK` on success.
ripoffline

Name
ripoffline — obtain a string of characters and their attributes from a curses window

Synopsis
#include <curses.h>
int ripoffline(int line, int (*init) (WINDOW *, int));

Description
The interface ripoffline() shall behave as specified in X/Open Curses, Issue 7, except that ripoffline() shall return -1 if the number of lines that were ripped off exceeds five.

scanw

Name
scanw — convert formatted input from a curses window

Synopsis
#include <curses.h>
int scanw(const char *fmt, ...);

Description
The scanw family of functions shall behave as described in X/Open Curses, Issue 7, except as noted below.

Differences
This function returns ERR on failure. On success it returns the number of successfully matched and assigned input items. This differs from X/Open Curses, Issue 7, which indicates this function returns OK on success.
vw_scanw

Name
vw_scanw — convert formatted input from a curses window

Synopsis
#include <curses.h>
int vw_scanw(WINDOW *win, const char *fmt, va_list vararglist);

Description
The scanw family of functions shall behave as described in X/Open Curses, Issue 7, except as noted below.

Differences
This function returns ERR on failure. On success it returns the number of successfully matched and assigned input items. This differs from X/Open Curses, Issue 7, which indicates this function returns OK on success.

vwscanw

Name
vwscanw — convert formatted input from a curses window

Synopsis
#include <curses.h>
int vw_scanw(WINDOW *win, const char *fmt, va_list vararglist);

Description
The scanw family of functions shall behave as described in X/Open Curses, Issue 7, except as noted below.

Differences
This function returns ERR on failure. On success it returns the number of successfully matched and assigned input items. This differs from X/Open Curses, Issue 7, which indicates this function returns OK on success.

winchnstr

Name
winchnstr — obtain a string of characters and their attributes from a curses window

Synopsis
#include <curses.h>
int winchnstr(WINDOW *win, chtype *chstr, int n);

Description
The interface winchnstr() shall behave as specified in X/Open Curses, Issue 7, except that winchnstr() shall return the number of characters that were read.
**winchstr**

**Name**

`winchstr` — obtain a string of characters and their attributes from a curses window

**Synopsis**

```c
#include <curses.h>
int winchstr(WINDOW * win, chtype * chstr);
```

**Description**

The interface `winchstr()` shall behave as specified in *X/Open Curses, Issue 7*, except that `winchstr()` shall return the number of characters that were read.

**winstr**

**Name**

`winstr` — obtain a string of characters from a curses window

**Synopsis**

```c
#include <curses.h>
int winstr(WINDOW * win, char * str);
```

**Description**

The interface `winstr()` shall behave as specified in *posix 1003.1-2008 (ISO/IEC 9945-2009)*, except that `winstr()` shall return the number of characters that were read.

**wscanw**

**Name**

`wscanw` — convert formatted input from a curses window

**Synopsis**

```c
#include <curses.h>
int wscanw(WINDOW * win, const char * fmt, ...);
```

**Description**

The `scanw` family of functions shall behave as described in *X/Open Curses, Issue 7*, except as noted below.

**Differences**

This function returns ERR on failure. On success it returns the number of successfully matched and assigned input items. This differs from *X/Open Curses, Issue 7*, which indicates this function returns OK on success.

**15.8 Interfaces for libncursesw**

`Table 15-7` defines the library name and shared object name for the libncursesw library
The behavior of the interfaces in this library is specified by the following specifications:

- [Libncursesw] Libncursesw API
- [LSB] This Specification
- [ncursesw] Libncursesw Placeholder
- [X-CURSES] X/Open Curses, Issue 7

### 15.8.1 Curses Wide

#### 15.8.1.1 Interfaces for Curses Wide

An LSB conforming implementation shall provide the generic functions for Curses Wide specified in Table 15-8, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>add_wch</td>
<td>Add an ASCII character to the screen.</td>
</tr>
<tr>
<td>add_wchnstr</td>
<td>Add an ASCII string to the screen.</td>
</tr>
<tr>
<td>add_wchstr</td>
<td>Add a wide character string to the screen.</td>
</tr>
<tr>
<td>addch</td>
<td>Add a single character to the screen.</td>
</tr>
<tr>
<td>addchnstr</td>
<td>Add a wide character string to the screen.</td>
</tr>
<tr>
<td>addchstr</td>
<td>Add a wide character string to the screen.</td>
</tr>
<tr>
<td>addstr</td>
<td>Add a string to the screen.</td>
</tr>
<tr>
<td>addwstr</td>
<td>Add a wide string to the screen.</td>
</tr>
<tr>
<td>assume_default_colors</td>
<td>Assume default colors.</td>
</tr>
<tr>
<td>attr_get</td>
<td>Get the current attribute.</td>
</tr>
<tr>
<td>attr_off</td>
<td>Turn off attributes.</td>
</tr>
<tr>
<td>attr_on</td>
<td>Turn on attributes.</td>
</tr>
<tr>
<td>attr_set</td>
<td>Set the current attribute.</td>
</tr>
<tr>
<td>atroff</td>
<td>Turn off attributes.</td>
</tr>
<tr>
<td>atron</td>
<td>Turn on attributes.</td>
</tr>
<tr>
<td>baudrate</td>
<td>Set baud rate.</td>
</tr>
<tr>
<td>beep</td>
<td>Beep the terminal.</td>
</tr>
<tr>
<td>bkgd</td>
<td>Set background color.</td>
</tr>
<tr>
<td>bkgdset</td>
<td>Set background color.</td>
</tr>
<tr>
<td>bkgrnd</td>
<td>Set background color.</td>
</tr>
<tr>
<td>bkgrndset</td>
<td>Set background color.</td>
</tr>
<tr>
<td>border</td>
<td>Set border.</td>
</tr>
<tr>
<td>border_set</td>
<td>Set border.</td>
</tr>
<tr>
<td>box</td>
<td>Set box.</td>
</tr>
<tr>
<td>box_set</td>
<td>Set box.</td>
</tr>
<tr>
<td>can_change_color</td>
<td>Can change color.</td>
</tr>
<tr>
<td>clear</td>
<td>Clear the screen.</td>
</tr>
<tr>
<td>color_set</td>
<td>Set color.</td>
</tr>
<tr>
<td>curs_set</td>
<td>Set curses.</td>
</tr>
<tr>
<td>curses_version</td>
<td>Set curses version.</td>
</tr>
<tr>
<td>def_prog_mode</td>
<td>Define program mode.</td>
</tr>
<tr>
<td>def_shell_mode</td>
<td>Define shell mode.</td>
</tr>
<tr>
<td>define_key</td>
<td>Define key.</td>
</tr>
<tr>
<td>del_curterm</td>
<td>Delete current term.</td>
</tr>
<tr>
<td>delay_output</td>
<td>Delay output.</td>
</tr>
<tr>
<td>delch</td>
<td>Delete character.</td>
</tr>
<tr>
<td>deleteln</td>
<td>Delete line.</td>
</tr>
<tr>
<td>doupdate</td>
<td>Dup date.</td>
</tr>
<tr>
<td>dupwin</td>
<td>Dup win.</td>
</tr>
<tr>
<td>echo</td>
<td>Echo character.</td>
</tr>
<tr>
<td>echo_wchar</td>
<td>Echo wide character.</td>
</tr>
<tr>
<td>echochar</td>
<td>Echo character.</td>
</tr>
<tr>
<td>endwin</td>
<td>End win.</td>
</tr>
<tr>
<td>erase</td>
<td>Erase character.</td>
</tr>
<tr>
<td>erasechar</td>
<td>Erase wide character.</td>
</tr>
<tr>
<td>filter</td>
<td>Filter.</td>
</tr>
<tr>
<td>flash</td>
<td>Flash.</td>
</tr>
<tr>
<td>flushinp</td>
<td>Flash input.</td>
</tr>
<tr>
<td>get_wch</td>
<td>Get a character.</td>
</tr>
<tr>
<td>get_wstr</td>
<td>Get a wide string.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>getbkgd</td>
<td>Get background color</td>
</tr>
<tr>
<td>getbgrnd</td>
<td>Get background color</td>
</tr>
<tr>
<td>getcchar</td>
<td>Get character</td>
</tr>
<tr>
<td>getch</td>
<td>Get character</td>
</tr>
<tr>
<td>getmouse</td>
<td>Get mouse</td>
</tr>
<tr>
<td>getn_wstr</td>
<td>Get n-width string</td>
</tr>
<tr>
<td>getnstr</td>
<td>Get n-string</td>
</tr>
<tr>
<td>getstr</td>
<td>Get string</td>
</tr>
<tr>
<td>getwin</td>
<td>Get window</td>
</tr>
<tr>
<td>halfdelay</td>
<td>Half delay</td>
</tr>
<tr>
<td>has_colors</td>
<td>Has colors</td>
</tr>
<tr>
<td>has_ic</td>
<td>Has ic</td>
</tr>
<tr>
<td>has_key</td>
<td>Has key</td>
</tr>
<tr>
<td>has_mouse</td>
<td>Has mouse</td>
</tr>
<tr>
<td>hline</td>
<td>Hline</td>
</tr>
<tr>
<td>idcok</td>
<td>Identify key</td>
</tr>
<tr>
<td>idlok</td>
<td>Identify key</td>
</tr>
<tr>
<td>immedok</td>
<td>Immediate delay</td>
</tr>
<tr>
<td>in_wch</td>
<td>In wch</td>
</tr>
<tr>
<td>in_wchnstr</td>
<td>In wch n-str</td>
</tr>
<tr>
<td>inch</td>
<td>Inch</td>
</tr>
<tr>
<td>inchnstr</td>
<td>Inch n-str</td>
</tr>
<tr>
<td>init_color</td>
<td>Initialize color</td>
</tr>
<tr>
<td>init_pair</td>
<td>Initialize pair</td>
</tr>
<tr>
<td>innstr</td>
<td>In n-str</td>
</tr>
<tr>
<td>ins_nwstr</td>
<td>In n-width n-str</td>
</tr>
<tr>
<td>ins_wch</td>
<td>In wch</td>
</tr>
<tr>
<td>ins_wstr</td>
<td>In wch n-str</td>
</tr>
<tr>
<td>insch</td>
<td>Insert character</td>
</tr>
<tr>
<td>insdellln</td>
<td>Insert dellln</td>
</tr>
<tr>
<td>insertln</td>
<td>Insert n-lee</td>
</tr>
<tr>
<td>intrflush</td>
<td>Insert flush</td>
</tr>
<tr>
<td>isendwin</td>
<td>Is endwin</td>
</tr>
<tr>
<td>key_name</td>
<td>Key name</td>
</tr>
<tr>
<td>keybound</td>
<td>Key bound</td>
</tr>
<tr>
<td>keyname</td>
<td>Key name</td>
</tr>
<tr>
<td>keypad</td>
<td>Keypad</td>
</tr>
<tr>
<td>killchar</td>
<td>Kill character</td>
</tr>
<tr>
<td>killwchar</td>
<td>Kill wchar</td>
</tr>
<tr>
<td>leaveok</td>
<td>Leave ok</td>
</tr>
<tr>
<td>longname</td>
<td>Long name</td>
</tr>
<tr>
<td>mcprint</td>
<td>Mcprint</td>
</tr>
<tr>
<td>move</td>
<td>Move</td>
</tr>
<tr>
<td>mouse_interval</td>
<td>Mouse interval</td>
</tr>
<tr>
<td>mousemask</td>
<td>Mouse mask</td>
</tr>
<tr>
<td>mvadd_wch</td>
<td>Move add wch</td>
</tr>
<tr>
<td>mvadd_wchnstr</td>
<td>Move add wch n-str</td>
</tr>
<tr>
<td>mvadd_wchstr</td>
<td>Move add wch str</td>
</tr>
<tr>
<td>mvaddch</td>
<td>Move add ch</td>
</tr>
<tr>
<td>mvaddchnstr</td>
<td>Move add ch n-str</td>
</tr>
<tr>
<td>mvaddnstr</td>
<td>Move add nstr</td>
</tr>
<tr>
<td>mvaddnstr</td>
<td>Move add n-str</td>
</tr>
<tr>
<td>mvaddwstr</td>
<td>Move add wstr</td>
</tr>
<tr>
<td>mvchgt</td>
<td>Move chgt</td>
</tr>
<tr>
<td>mvcurs</td>
<td>Move curs</td>
</tr>
<tr>
<td>mvderwin</td>
<td>Move derivative win</td>
</tr>
<tr>
<td>mvgetch</td>
<td>Move getch</td>
</tr>
<tr>
<td>mvgetn_wstr</td>
<td>Move get n-wstr</td>
</tr>
<tr>
<td>mvgetnstr</td>
<td>Move get n-str</td>
</tr>
<tr>
<td>mvgetwch</td>
<td>Move get wch</td>
</tr>
<tr>
<td>mvget_wstr</td>
<td>Move get w-str</td>
</tr>
<tr>
<td>mvhline</td>
<td>Move hline</td>
</tr>
<tr>
<td>mvhline_set</td>
<td>Move hline set</td>
</tr>
<tr>
<td>mvin_wch</td>
<td>Move in wch</td>
</tr>
<tr>
<td>mvin_wchnstr</td>
<td>Move in wch n-str</td>
</tr>
<tr>
<td>mvinch</td>
<td>Move in ch</td>
</tr>
<tr>
<td>mvinchnstr</td>
<td>Move in ch n-str</td>
</tr>
<tr>
<td>mvinnstr</td>
<td>Move in nstr</td>
</tr>
<tr>
<td>mvinnwstr</td>
<td>Move in n-wstr</td>
</tr>
<tr>
<td>mvins_nwstr</td>
<td>Move in n-width n-str</td>
</tr>
<tr>
<td>mvins_wch</td>
<td>Move in wch</td>
</tr>
<tr>
<td>mvins_wstr</td>
<td>Move in w-str</td>
</tr>
<tr>
<td>mvinschn</td>
<td>Move in chn</td>
</tr>
<tr>
<td>mvinsnstr</td>
<td>Move in nstr</td>
</tr>
<tr>
<td>mvinsnstr</td>
<td>Move in nstr</td>
</tr>
<tr>
<td>mvintr</td>
<td>Move in intr</td>
</tr>
<tr>
<td>mvintrwstr</td>
<td>Move in intr wstr</td>
</tr>
<tr>
<td>mvprintw</td>
<td>Move printw</td>
</tr>
<tr>
<td>mvscanw</td>
<td>Move scanw</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>mvvline</td>
<td>Move vertical line</td>
</tr>
<tr>
<td>mvvline_set</td>
<td>Move vertical line set</td>
</tr>
<tr>
<td>mvwadd_wch</td>
<td>Move vertical add wch</td>
</tr>
<tr>
<td>mvwadd_wchnstr</td>
<td>Move vertical add wchnstr</td>
</tr>
<tr>
<td>mvwaddnstr</td>
<td>Move vertical add nstr</td>
</tr>
<tr>
<td>mvwchcat</td>
<td>Move virtual charcat</td>
</tr>
<tr>
<td>mvwdelch</td>
<td>Move virtual delete char</td>
</tr>
<tr>
<td>mvwget_wch</td>
<td>Move virtual get wch</td>
</tr>
<tr>
<td>mvwget_wchstr</td>
<td>Move virtual get wch str</td>
</tr>
<tr>
<td>mvwline</td>
<td>Move virtual line</td>
</tr>
<tr>
<td>mvwline_set</td>
<td>Move virtual line set</td>
</tr>
<tr>
<td>mvwin</td>
<td>Move virtual win</td>
</tr>
<tr>
<td>mvwin_wchstr</td>
<td>Move virtual win wch str</td>
</tr>
<tr>
<td>mvwinwstr</td>
<td>Move virtual win wstr</td>
</tr>
<tr>
<td>mvwins_wch</td>
<td>Move virtual ins wch</td>
</tr>
<tr>
<td>mvwins_wstr</td>
<td>Move virtual ins wstr</td>
</tr>
<tr>
<td>mvwprintw</td>
<td>Move vertical print wstr</td>
</tr>
<tr>
<td>mvwscanw</td>
<td>Move virtual scanw</td>
</tr>
<tr>
<td>newpad</td>
<td>New pad</td>
</tr>
<tr>
<td>nobreak</td>
<td>No break</td>
</tr>
<tr>
<td>nocbreak</td>
<td>No cbreak</td>
</tr>
<tr>
<td>noqiflush</td>
<td>No qiflush</td>
</tr>
<tr>
<td>nooverwrite</td>
<td>No overwrite</td>
</tr>
<tr>
<td>noqiflush</td>
<td>No qiflush</td>
</tr>
<tr>
<td>overwrite</td>
<td>Over write</td>
</tr>
<tr>
<td>npoutrefresh</td>
<td>New pout refresh</td>
</tr>
<tr>
<td>putwin</td>
<td>Put win</td>
</tr>
<tr>
<td>refresh</td>
<td>Refresh</td>
</tr>
<tr>
<td>resizeterm</td>
<td>Resize term</td>
</tr>
<tr>
<td>scanw</td>
<td>Scan w</td>
</tr>
<tr>
<td>scr_set</td>
<td>Screen set</td>
</tr>
<tr>
<td>set_curterm</td>
<td>Set current term</td>
</tr>
<tr>
<td>setupterm</td>
<td>Set up term</td>
</tr>
<tr>
<td>slk_attr</td>
<td>Set line key attr</td>
</tr>
<tr>
<td>slk_attr_set</td>
<td>Set line key attr set</td>
</tr>
<tr>
<td>slk_clear</td>
<td>Set line key clear</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>slk_noutrefresh</td>
<td>slk_refresh</td>
</tr>
<tr>
<td>slk_refresh</td>
<td>slk_restore</td>
</tr>
<tr>
<td>slk_set</td>
<td></td>
</tr>
<tr>
<td>start_color</td>
<td>subpad</td>
</tr>
<tr>
<td>standend</td>
<td>subwin</td>
</tr>
<tr>
<td>standout</td>
<td></td>
</tr>
<tr>
<td>termattrs [X-CURSES]</td>
<td>termattrs</td>
</tr>
<tr>
<td>tgetflag [Libncursesw]</td>
<td>tigetflag</td>
</tr>
<tr>
<td>tgetflag</td>
<td></td>
</tr>
<tr>
<td>touchline [Libncursesw]</td>
<td>touchwin</td>
</tr>
<tr>
<td>typeahead [Libncursesw]</td>
<td>unctrl</td>
</tr>
<tr>
<td>ungetmouse [Libncursesw]</td>
<td>untouchwin</td>
</tr>
<tr>
<td>use_extended_names [Libncursesw]</td>
<td>vid_attr [X-CURSES]</td>
</tr>
<tr>
<td>use_extended_names [Libncursesw]</td>
<td>vid_puts</td>
</tr>
<tr>
<td>vidputs [Libncursesw]</td>
<td>vline</td>
</tr>
<tr>
<td>vw_scanw [Libncursesw]</td>
<td>vwprintw</td>
</tr>
<tr>
<td>wadd_wchstr [Libncursesw]</td>
<td>wadd_wchstr</td>
</tr>
<tr>
<td>waddchstr [Libncursesw]</td>
<td>waddnstr</td>
</tr>
<tr>
<td>waddwstr [Libncursesw]</td>
<td>wattr_get</td>
</tr>
<tr>
<td>wattr_set [Libncursesw]</td>
<td>wattr_get_off</td>
</tr>
<tr>
<td>wbkgd [Libncursesw]</td>
<td>wbkgrpset</td>
</tr>
<tr>
<td>wborder [Libncursesw]</td>
<td>wborder_set</td>
</tr>
<tr>
<td>wclrrobot [Libncursesw]</td>
<td>wclrrobot</td>
</tr>
<tr>
<td>wdnoch [Libncursesw]</td>
<td>wdeleteeln</td>
</tr>
<tr>
<td>werase [Libncursesw]</td>
<td>wget_wch</td>
</tr>
<tr>
<td>wgetch [Libncursesw]</td>
<td>wgetn_wstr</td>
</tr>
<tr>
<td>whline [Libncursesw]</td>
<td>whline_set</td>
</tr>
<tr>
<td>win_wchstr [Libncursesw]</td>
<td>winch</td>
</tr>
<tr>
<td>winnstr [Libncursesw]</td>
<td>winnstr</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
15 Utility Libraries

LSB Core - Generic 5.0

<table>
<thead>
<tr>
<th>[Libncursesw]</th>
<th>[Libncursesw]</th>
<th>[Libncursesw]</th>
<th>[Libncursesw]</th>
</tr>
</thead>
<tbody>
<tr>
<td>wins_wstr</td>
<td>winsch</td>
<td>winsdelln</td>
<td>winsertln</td>
</tr>
<tr>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
</tr>
<tr>
<td>winsnr</td>
<td>winsstr</td>
<td>winstr</td>
<td>winstr</td>
</tr>
<tr>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
</tr>
<tr>
<td>wmousc_trafo</td>
<td>wmove</td>
<td>wnoutrefresh</td>
<td>wprinter</td>
</tr>
<tr>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
</tr>
<tr>
<td>wdrawln</td>
<td>wrefresh</td>
<td>wresize</td>
<td>wscanw</td>
</tr>
<tr>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
</tr>
<tr>
<td>wscl</td>
<td>wsetscrreg</td>
<td>wstandend</td>
<td>wstandout</td>
</tr>
<tr>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
</tr>
<tr>
<td>wscopydown</td>
<td>wsyncup</td>
<td>wtimeout</td>
<td>wtouchln</td>
</tr>
<tr>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
</tr>
<tr>
<td>wunctrl</td>
<td>wvline</td>
<td>wvline_set</td>
<td></td>
</tr>
<tr>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic deprecated functions for Curses Wide specified in Table 15-9, 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 15-9 libncursesw - Curses Wide Deprecated Function Interfaces**

<table>
<thead>
<tr>
<th>tgetent</th>
<th>tgetflag</th>
<th>tgetnum</th>
<th>tgetstr</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
<td>[Libncursesw]</td>
</tr>
<tr>
<td>tgoto</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[Libncursesw]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Curses Wide specified in Table 15-10, with the full mandatory functionality as described in the referenced underlying specification.

**Table 15-10 libncursesw - Curses Wide Data Interfaces**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>stdscr [ncursesw]</td>
<td>ttytype [ncursesw]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

15.9 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.

15.9.1 ncursesw/curses.h

```c
#define CURSES 1
#define setsyx(y,x) do { if (newscr) { 
    if ((y) == -1 && (x) == -1) 
      leaveok(newscr, TRUE); 
    else { 
      leaveok(newscr, FALSE); 
      wmove(newscr, (y), (x)); 
    } 
  } while(0)
#define getsyx(y,x) do { if (newscr) { 
    if (is_leaveok(newscr)) 
      (y) = (x) = -1; 
    else 
      getyx(newscr,(y), (x)); 
  } while(0)
#define CURSES_H 1
#define NCURSES_VERSION_MAJOR 5
#define NCURSES_VERSION_MINOR 9
#define NCURSES_VERSION_PATCH 20110404
#define NCURSES_VERSION "5.9"
#define NCURSES_MOUSE_VERSION 1
#define NCURSES_ENABLE_STDBOOL_H 1
#define NCURSES_INLINE inline
#define NCURSES_TPARM_VARARGS 1
#ifndef TRUE
#define TRUE 1
#endif
#define NCURSES_BOOL bool
#ifdef __cplusplus
#  define NCURSES_CAST(type,value) static_cast<type>(value)
#else
#  define NCURSES_CAST(type,value) (type)(value)
#endif
#define WA_ATTRIBUTES A_ATTRIBUTES
#define WA_NORMAL A_NORMAL
#define WA_STANDOUT A_STANDOUT
#define WA_UNDERLINE A_UNDERLINE
#define WA_REVERSE A_REVERSE
#define WA_BLINK A_BLINK
#define WA_DIM A_DIM
#define WA_BOLD A_BOLD
#define WA_ALTCARSET A_ALTCARSET
#define WA_INVIS A_INVIS
#define WA_PROTECT A_PROTECT
#define WA_HORIZONTAL A_HORIZONTAL
#define WA_LEFT A_LEFT
#define WA_LOW A_LOW
#define WA_RIGHT A_RIGHT
#define WA_TOP A_TOP
#define WA_VERTICAL A_VERTICAL
#define COLOR_BLACK 0
#define COLOR_RED 1
#define COLOR_GREEN 2
#define COLOR_YELLOW 3
```

```c
#define COLOR_BLUE      4
#define COLOR_MAGENTA   5
#define COLOR_CYAN      6
#define COLOR_WHITE     7
#define NCURSES_ACS(c) (acs_map[NCURSES_CAST(unsigned char,c)])
#define ACS_ULCORNER    NCURSES_ACS('l')
#define ACS_LLCORNER    NCURSES_ACS('m')
#define ACS_URCORNER    NCURSES_ACS('k')
#define ACS_LRCORNER    NCURSES_ACS('j')
#define ACS_LTEE        NCURSES_ACS('t')
#define ACS_RTEE        NCURSES_ACS('u')
#define ACS_BTEE        NCURSES_ACS('v')
#define ACS_TTEE        NCURSES_ACS('w')
#define ACS_HLINE       NCURSES_ACS('q')
#define ACS_VLINE       NCURSES_ACS('x')
#define ACS_PLUS        NCURSES_ACS('n')
#define ACS_S1  NCURSES_ACS('o')
#define ACS_S9  NCURSES_ACS('s')
#define ACS_DIAMOND     NCURSES_ACS('`')
#define ACS_CKBOARD     NCURSES_ACS('a')
#define ACS_DEGREE      NCURSES_ACS('f')
#define ACS_PLMINUS     NCURSES_ACS('g')
#define ACS_BULLET      NCURSES_ACS('~')
#define ACS_LARROW      NCURSES_ACS(',')
#define ACS_RARROW      NCURSES_ACS('+')
#define ACS_DARROW      NCURSES_ACS('.')
#define ACS_UARROW      NCURSES_ACS('-')
#define ACS_BOARD       NCURSES_ACS('h')
#define ACS_LANTERN     NCURSES_ACS('i')
#define ACS_BLOCK       NCURSES_ACS('0')
#define ACS_S3  NCURSES_ACS('p')
#define ACS_S7  NCURSES_ACS('r')
#define ACS_LEQUAL      NCURSES_ACS('y')
#define ACS_GEQUAL      NCURSES_ACS('z')
#define ACS_PI  NCURSES_ACS('{')
#define ACS_NEQUAL      NCURSES_ACS('|')
#define ACS_STERLING    NCURSES_ACS('}')
#define ACS_BSSB        ACS_ULCORNER
#define ACS_SSSB        ACS_LLCORNER
#define ACS_BBSS        ACS_URCORNER
#define ACS_SBBSS       ACS_LRCORNER
#define ACS_BSSSS       ACS_RTEE
#define ACS_SBSB        ACS_LTEE
#define ACS_SSSS        ACS_BTEE
#define ACS_WSBS        ACS_HLINE
#define ACS_SBSB        ACS_VLINE
#define ACS_WARNING     ACS_PLUS
#define ERR     (-1)
#define OK      (0)
#define _SUBWIN 0x01
#define _ENDLINE 0x02
#define _FULLWIN 0x04
#define _SCROLLWIN 0x08
#define _ISPAD 0x10
#define _HASMOVED 0x20
#define _WRAPPED 0x40
#define _NOCHANGE -1
#define _NEWINDEX -1
#define CCHARW_MAX      5
#define NCURSES_EXT_COLORS 20110404
#define GCC_PRINTFLIKE(fmt,var) __attribute__((format(printf,fmt,var)))
#define GCC_SCANFLIKE(fmt,var) __attribute__((format(scanf,fmt,var)))
#define NCURSES_EXT_FUNCS 20110404
```
#define curses_version() NCURSES_VERSION
#define NCURSES_SP_FUNCS 281104084
#define NCURSES_SP_OUTC NCURSES_SP_NAME(NCURSES_OUTC)
#define NCURSES_SP_NAME(name) name
#define NCURSES_ATTR_SHIFT 8
#define NCURSES_BITS(mask, shift) (((mask) << ((shift) + NCURSES_ATTR_SHIFT))
#define A_NORMAL (1UL - 1UL)
#define A_ATTRIBUTES NCURSES_BITS(-1UL - 1UL, 0)
#define A_CHARS TEXT (NCURSES_BITS(1UL, 0) - 1UL)
#define A_COLOR NCURSES_BITS((1UL) << 8 - 1UL, 0)
#define A_STANDOUT NCURSES_BITS(1UL, 8)
#define A_UNDERLINE NCURSES_BITS(1UL, 9)
#define A_REVERSE NCURSES_BITS(1UL, 10)
#define A_BOLD NCURSES_BITS(1UL, 11)
#define A_DIM NCURSES_BITS(1UL, 12)
#define A_BOLD NCURSES_BITS(1UL, 13)
#define A_ALTCHARSET NCURSES_BITS(1UL, 14)
#define A_INVIS NCURSES_BITS(1UL, 15)
#define A_PROTECT NCURSES_BITS(1UL, 16)
#define A_HORIZONTA NCURSES_BITS(1UL, 17)
#define A_LEFT NCURSES_BITS(1UL, 18)
#define A_LOW NCURSES_BITS(1UL, 19)
#define A_RIGHT NCURSES_BITS(1UL, 20)
#define A_TOP NCURSES_BITS(1UL, 21)
#define A_VERTI NCURSES_BITS(1UL, 22)
#define getyx(win, y, x) (y = getcury(win), x = getcurx(win))
#define getbegyx(win, y, x) (y = getbegy(win), x = getbegx(win))
#define getmaxyx(win, y, x) (y = getmaxy(win), x = getmaxx(win))
#define getparyx(win, y, x) (y = getpary(win), x = getparx(win))
#define wgetstr(w, s) wgetnstr(w, s, -1)
#define getnstr(s, n) wgetnstr(stdscr, s, n)
#define setterm(term) setupterm(term, 1, (int *)0)
#define fixterm() reset_prog_mode()
#define resetterm() reset_shell_mode()
#define saveterm() def_prog_mode()
#define crmode() cbreak()
#define nocrmode() nocbreak()
#define getattrs(win) NCURSES_CAST(int, (win) ? (win)->_attrs : A_NORMAL)
#define getcurx(win) ((win) ? (win)->_curx : ERR)
#define getcury(win) ((win) ? (win)->_cury : ERR)
#define getbegx(win) ((win) ? (win)->_begx : ERR)
#define getbegy(win) ((win) ? (win)->_begy : ERR)
#define getmaxx(win) ((win) ? ((win)->_maxx + 1) : ERR)
#define getmaxy(win) ((win) ? ((win)->_maxy + 1) : ERR)
#define getparx(win) ((win) ? (win)->_parx : ERR)
#define getpary(win) ((win) ? (win)->_pary : ERR)
#define wstandout(win) (wattrset(win, A_STANDOUT))
#define wstandend(win) (wattrset(win, A_NORMAL))
#define wattron(win, at) wattr_on(win, NCURSES_CAST(attr_t, at), NULL)
#define wattroff(win, at) wattr_off(win, NCURSES_CAST(attr_t, at), NULL)
#define scroll(win) wscrl(win, 1)
#define touchwin(win) wtouchn((win), 0, getmaxy(win), 1)
#define touchline(win, s, c) wtouchn((win), s, c, 1)
#define untouchwin(win) wtouchn((win), 0, getmaxy(win), 0)
#define border(ls, rs, ts, bs, tl, tr, bl, br) wborder(stdscr, ls, rs, ts, bs, tl, tr, bl, br)
#define hline(ch, n) whline(stdscr, ch, n)
#define vline(ch, n) wvline(stdscr, ch, n)
15 Utility Libraries

__LSB Core - Generic 5.0__

```c
#define winstr(w, s)    winnstr(w, s, -1)
#define winchstr(w, s)  winchnstr(w, s, -1)
#define wnsstr(w, s)    wnsnstr(w, s, -1)
#define redrawwin(win)  wredrawln(win, 0, (win)->_maxy+1)
#define waddstr(win, str)  waddnstr(win, str, -1)
#define waddchstr(win, str)  waddchnstr(win, str, -1)
#define COLOR_PAIR(n)   NCURSES_BITS(n, 0)
#define PAIR_NUMBER(a)  (NCURSES_CAST(int,((NCURSES_CAST(unsigned
long, a) & A_COLOR) >> NCURSES_ATTR_SHIFT)))
#define addch(ch)       waddch(stdscr, ch)
#define addchnstr(str, n)  waddchnstr(stdscr, str, n)
#define addchstr(str)    waddchstr(stdscr, str)
#define addnstr(str, n)  waddnstr(stdscr, str, n)
#define addstr(str)      waddnstr(stdscr, str, -1)
#define attroff(at)      wattroff(stdscr, at)
#define attron(at)       wattron(stdscr, at)
#define attrset(at)      wattrset(stdscr, at)
#define attr_get(ap, cp, o)       wattr_get(stdscr, ap, cp, o)
#define attr_off(a, o)    wattr_off(stdscr, a, o)
#define attr_on(a, o)     wattr_on(stdscr, a, o)
#define attr_set(a, c, o) wattr_set(stdscr, a, c, o)
#define bkgd(ch)         wbkgd(stdscr, ch)
#define bkgdset(ch)       wbkgdset(stdscr, ch)
#define chgat(n, a, c, o) wchgat(stdscr, n, a, c, o)
#define clear()          wclear(stdscr)
#define clrtobot()        wclrtobot(stdscr)
#define clrtoeol()        wclrtoeol(stdscr)
#define color_set(c, o)   wcolor_set(stdscr, c, o)
#define delch()          wdelch(stdscr)
#define deletein()       winsdelln(stdscr, -1)
#define echochar(c)       wechochar(stdscr, c)
#define erase()          werase(stdscr)
#define getch()          wgetch(stdscr)
#define getstr(str)       wgetstr(stdscr, str)
#define inch()           winch(stdscr)
#define inchstr(s, n)     winchnstr(stdscr, s, n)
#define inchnstr(s)       winchnstr(stdscr, s)
#define insch(c)          winsch(stdscr, c)
#define insdelln(n)       winsdelln(stdscr, n, s, n)
#define insertln()       winsdelln(stdscr, 1)
#define insnstr(s, n)     winsnstr(stdscr, s, n)
#define insstr(s)         winsstr(stdscr, s)
#define instr(s)          winstr(stdscr, s)
#define move(y, x)        wmove(stdscr, y, x)
#define refresh()         wrefresh(stdscr)
#define scrl(n)           wscroll(stdscr, n)
#define setscreg(t, b)    wsetscregs(stdscr, t, b)
#define standend()        wstandend(stdscr)
#define standout()        wstandout(stdscr)
#define timeout(delay)    wtimeout(stdscr, delay)
#define wdeleteln(win)    winsdelln(win, -1)
#define winsertln(win)    winsdelln(win, 1)
#define mvwaddch(win, y, x, ch)    (wmove(win, y, x) == ERR ? ERR : waddch(win, ch))
#define mvwaddchnstr(win, y, x, str, n)  (wmove(win, y, x) == ERR ? ERR : waddchnstr(win, str, n))
#define mvwaddchstr(win, y, x, str)     (wmove(win, y, x) == ERR ? ERR : waddchstr(win, str))
#define mvwaddnstr(win, y, x, str, -1)) (wmove(win, y, x) == ERR ? ERR : waddnstr(win, str, -1))
#define mvwaddstr(win, y, x, str)       (wmove(win, y, x) == ERR ? ERR : waddstr(win, str))
#define mvwaddchstr(win, y, x, str)     (wmove(win, y, x) == ERR ? ERR : waddchnstr(win, str, -1))
#define mvwaddstr(win, y, x, str)       (wmove(win, y, x) == ERR ? ERR : waddstr(win, str, -1))
#define mvwaddstr(win, y, x, str)       (wmove(win, y, x) == ERR ? ERR : waddstr(win, str, -1))
#define mvwaddstr(win, y, x, str)       (wmove(win, y, x) == ERR ? ERR : waddstr(win, str, -1))
```
ERR : wchgat(win,n,a,c,o))
define mwwgetch(win,y,x)     (wmvgetch(win,y,x) == ERR ? ERR : wgetch(win))
define mwwgetnstr(win,y,x,str,n)   (wmvgetnstr(win,y,x,str,n) == ERR ? ERR : wgetnstr(win,str,n))
define mwwgetstr(win,y,x,str)  (wmvwgetstr(win,y,x,str) == ERR ? ERR : wgetstr(win,str))
define mwwhline(win,y,x,c,n)   (wmvwhline(win,y,x,c,n) == ERR ? ERR : whline(win,c,n))
define mwwinch(win,y,x)       (wmvwinch(win,y,x) == ERR ? ERR : winch(win))
define mwgetnstr(win,y,x,str,n)  (wmvgetnstr(win,y,x,str,n) == ERR ? ERR : wgetnstr(win,str,n))
define mwwgetstr(win,y,x,str)  (wmvwgetstr(win,y,x,str) == ERR ? ERR : wgetstr(win,str))
define mwvline(line,y,x,c,n)   (wmvline(line,y,x,c,n) == ERR ? ERR : vline(line,c,n))
define mvaddch(y,x,ch) mvwaddch(stdscr,y,x,ch)
define mvaddchnstr(y,x,str,n)  mvwaddchnstr(stdscr,y,x,str,n)
define mvaddstr(y,x,str)      mvwaddstr(stdscr,y,x,str)
define mvaddnstr(y,x,str,n)    mvwaddnstr(stdscr,y,x,str,n)
define mvaddstr(y,x,str)      mvwaddstr(stdscr,y,x,str)
define mvdelch(y,x)    mvwdelch(stdscr,y,x)
define mvgetch(y,x)    mwwgetch(stdscr,y,x)
define mvgetnstr(y,x,str,n)  mwwgetnstr(stdscr,y,x,str,n)
define mvinch(y,x)     mwvinch(stdscr,y,x)
define mvinchnstr(y,x,s,n)  mwvinchnstr(stdscr,y,x,s,n)
define mvwinsch(y,x,c)   mwwinsch(stdscr,y,x,c)
define mvwinsnstr(y,x,s,n) mwwinsnstr(stdscr,y,x,s,n)
define mvwinsstr(y,x,s)   mwwinsstr(stdscr,y,x,s)
define mvwinstr(y,x,s)   mwwinstr(stdscr,y,x,s)
define mvvline(y,x,c,n)   mvvline(stdscr,y,x,c,n)
define getbkgd(win)    ((win)->_bkgd)
define slk_attr_off(a,v)       ((v) ? ERR : slk_attroff(a))
define slk_attr_on(a,v)        ((v) ? ERR : slk_attron(a))
define wattr_set(win,a,p,opts) (wattr_set(win,a,p,opts))
define vwprintw      vwprintw
#define vw_scanw      vwscanw
#define vsscanf(a,b,c) _nc_vsscanf(a,b,c)
define is_cleared(win)  ((win) ? (win)->_clear : FALSE)
define is_idcok(win)   ((win) ? (win)->_idcok : FALSE)
define is_idlok(win)   ((win) ? (win)->_idlok : FALSE)
define is_immedok(win)  ((win) ? (win)->_immed : FALSE)
define is_keypad(win)  ((win) ? (win)->_use_keypad : FALSE)
define is_leaveok(win)  ((win) ? (win)->_leaveok : FALSE)
define is_nodelay(win)  ((win) ? (win)->_delay : FALSE)
define is_notimeout(win) ((win) ? (win)->_notimeout : FALSE)
define is_pad(win)     ((win) ? (win)->_flags & _ISPAD) != 0 : FALSE)
#define is_scrollok(win)        ((win) ? (win)->_scroll : FALSE)
#define is_subwin(win)  ((win) ? ((win)->_flags & _SUBWIN) != 0 : FALSE)
#define is_syncok(win)  ((win) ? (win)->_sync : FALSE)
#define wgetparent(win) ((win) ? (win)->_parent : 0)
#define wgetscrreg(win,t,b)     ((win) ? (*(t) = (win)->_regtop, *(b) = (win)->_regbottom, OK) : ERR)
#define KEY_CODE_YES    0400
#define KEY_MIN 0401
#define KEY_BREAK 0401
#define KEY_SRESET  0530
#define KEY_RESET   0531
#define KEY_DOWN    0402
#define KEY_UP      0403
#define KEY_LEFT    0404
#define KEY_RIGHT   0405
#define KEY_HOME    0406
#define KEY_BACKSPACE 0407
#define KEY_F0  0410
#define KEY_DL 0510
#define KEY_IL 0511
#define KEY_DC  0512
#define KEY_IC  0513
#define KEY_EIC 0514
#define KEY_CLEAR  0515
#define KEY_EOL 0516
#define KEY_EOS 0517
#define KEY_SF  0520
#define KEY_SR 0521
#define KEY_NPAGE 0522
#define KEY_PPAGE 0523
#define KEY_STAB 0524
#define KEY_CTAB 0525
#define KEY_CATAB 0526
#define KEY_ENTER 0527
#define KEY_PRINT 0528
#define KEY_LL 0530
#define KEY_A1 0531
#define KEY_A3 0532
#define KEY_B2 0533
#define KEY_C1 0534
#define KEY_C3 0540
#define KEY_BTAB 0541
#define KEY_BEG 0542
#define KEY_CANCEL 0543
#define KEY_CLOSE 0544
#define KEY_COMMAND 0545
#define KEY_COPY 0546
#define KEY_CREATE 0547
#define KEY_END 0550
#define KEY_EXIT 0551
#define KEY_FIND 0552
#define KEY_HELP 0553
#define KEY_MARK 0554
#define KEY_MESSAGE 0555
#define KEY_MOVE 0556
#define KEY_NEXT 0557
#define KEY_OPEN 0558
#define KEY_OPTIONS 0560
#define KEY_PPAGE 0561
#define KEY_PREVIOUS 0562
#define KEY_REDO 0563
#define KEY_REFRESH 0564
#define KEY_REFERENCE 0565
#define KEY_REPLACE 0566
#define KEY_RESTART 0567
```c
#define KEY_RESUME 0570
#define KEY_SAVE 0571
#define KEY_SBEG 0572
#define KEY_SCANCEL 0573
#define KEY_SCOMMAND 0574
#define KEY_SCOPY 0575
#define KEY_SCREATE 0576
#define KEY_SDC 0577
#define KEY_SDL 0600
#define KEY_SELECT 0601
#define KEY_SEND 0602
#define KEY_SEOL 0603
#define KEY_SEXIT 0604
#define KEY_SFIND 0605
#define KEY_SHOME 0606
#define KEY_SHOME 0607
#define KEY_SIC 0610
#define KEY_SLEFT 0611
#define KEY_SMMESSAGE 0612
#define KEY_SMOVE 0613
#define KEY_SNEXT 0614
#define KEY_SOPTIONS 0615
#define KEY_SPREVIOUS 0616
#define KEY_SPRINT 0617
#define KEY_SREDO 0620
#define KEY_SREPLACE 0621
#define KEY_SRIGHT 0622
#define KEY_SRSUME 0623
#define KEY_SSAVE 0624
#define KEY_SSUSPEND 0625
#define KEY_UNDO 0626
#define KEY_SUTILITY 0627
#define KEY_SUSPEND 0628
#define KEY_UNDO 0629
#define KEY_MOUSE 0630
#define KEY_RESET 0631
#define KEY_EVENT 0632
#define KEY_MAX 0777
#define _XOPEN_CURSES 1
#define NCURSES_WACS(c) (&_nc_wacs[(unsigned char)c])
#define WACS_BSSB NCURSES_WACS('l')
#define WACS_SSBB NCURSES_WACS('m')
#define WACS_BSSS NCURSES_WACS('k')
#define WACS_SBBS NCURSES_WACS('j')
#define WACS_SBSS NCURSES_WACS('u')
#define WACS_SSSB NCURSES_WACS('t')
#define WACS_SSBS NCURSES_WACS('v')
#define WACS_BSSS NCURSES_WACS('w')
#define WACS_SSBS NCURSES_WACS('q')
#define WACS_SBSB NCURSES_WACS('x')
#define WACS_SSSS NCURSES_WACS('n')
#define WACS_ULCORNER WACS_BSSB
#define WACS_URCORNER WACS_SSBB
#define WACS_ULCORNER WACS_BBSS
#define WACS_URCORNER WACS_BBSS
#define WACS_LCORNER WACS_SBBS
#define WACS_LRCORNER WACS_SBSB
#define WACS_RTEE WACS_SBSS
#define WACS_LTEE WACS_SSSB
#define WACS_BTEE WACS_SSBS
#define WACS_TTEE WACS_BBSS
#define WACS_HLINE WACS_BBSS
#define WACS_VLINE WACS_SBSS
#define WACS_PLUS WACS_SBSS
#define WACS_S1 NCURSES_WACS('o')
#define WACS_S9 NCURSES_WACS('s')
#define WACS_DIAMOND NCURSES_WACS('`')
#define WACS_CKBOARD NCURSES_WACS('a')
#define WACS_DEGREE NCURSES_WACS('f')
``
```c
#define WACS_PLMINUS NCURSES_WACS('g')
#define WACS_BULLET NCURSES_WACS('~')
#define WACS_LARROW NCURSES_WACS(',')
#define WACS_RARROW NCURSES_WACS('+')
#define WACS_DARROW NCURSES_WACS('.')
#define WACS_UARROW NCURSES_WACS('-')
#define WACS_BOARD NCURSES_WACS('h')
#define WACS_LANTERN NCURSES_WACS('i')
#define WACS_BLOCK NCURSES_WACS('0')
#define WACS_S3 NCURSES_WACS('p')
#define WACS_S7 NCURSES_WACS('r')
#define WACS_LEQUAL NCURSES_WACS('y')
#define WACS_GEQUAL NCURSES_WACS('z')
#define WACS_PI NCURSES_WACS('{')
#define WACS_NEQUAL NCURSES_WACS('|')
#define WACS_STERLING NCURSES_WACS('}')
#define WACS_BDDB NCURSES_WACS('C')
#define WACS_DDBB NCURSES_WACS('D')
#define WACS_BBDD NCURSES_WACS('B')
#define WACS_DBBD NCURSES_WACS('A')
#define WACS_DDDB NCURSES_WACS('G')
#define WACS_DDDB NCURSES_WACS('F')
#define WACS_DDBD NCURSES_WACS('H')
#define WACS_BDDD NCURSES_WACS('I')
#define WACS_BDBD NCURSES_WACS('R')
#define WACS_DBDB NCURSES_WACS('Y')
#define WACS_DDDD NCURSES_WACS('E')
#define WACS_D_ULCORNER WACS_BDDB
#define WACS_D_LLCORNER WACS_DDBB
#define WACS_D_UNCORNER WACS_BBDD
#define WACS_D_LRCORNER WACS_DDBB
#define WACS_D_RTEE WACS_DBBD
#define WACS_D_BTEE WACS_DDBD
#define WACS_D_TTEE WACS_BDDD
#define WACS_D_HLINE WACS_BDBD
#define WACS_D_VLINE WACS_DBDB
#define WACS_D_PLUS WACS_DDDD
#define WACS_BTTB NCURSES_WACS('L')
#define WACS_TTBW NCURSES_WACS('M')
#define WACS_BBTW NCURSES_WACS('K')
#define WACS_TTBW NCURSES_WACS('J')
#define WACS_TBTT NCURSES_WACS('U')
#define WACS_TTTB NCURSES_WACS('T')
#define WACS_TTBW NCURSES_WACS('V')
#define WACS_BTBT NCURSES_WACS('W')
#define WACS_BBTB NCURSES_WACS('Q')
#define WACS_TBTT NCURSES_WACS('X')
#define WACS_TTTT NCURSES_WACS('N')
#define WACS_T_ULCORNER WACS_BTBB
#define WACS_T_LLCORNER WACS_TTBB
#define WACS_T_UNCORNER WACS_BBBT
#define WACS_T_LRCORNER WACS_TBTT
#define WACS_T_RTEE WACS_TBTT
#define WACS_T_BTEE WACS_TBTT
#define WACS_T_TTEE WACS_BTBT
#define WACS_T_HLINE WACS_BTBT
#define WACS_T_VLINE WACS_TBTT
#define WACS_T_PLUS WACS_TTTT
#define add_wch(c) wadd_wch(stdscr,c)
#define add_wchstr(str,n) wadd_wchstr(stdscr,str,n)
#define addwchstr(str) wadd_wchstr(stdscr,str)
#define addnwstr(wstr,n) waddnwstr(stdscr,wstr,n)
#define addwstr(wstr) waddwstr(stdscr,wstr)
#define bkgrnd(c) wbkgrnd(stdscr,c)
```
#define bkgrndset(c)    wbkgrndset(stdscr,c)
#define border_set(l,r,t,b,tl,tr,bl,br)    wborder_set(l,r,t,b,tl,tr,bl,br)
#define echo_wchar(c)  wecho_wchar(stdscr,c)
#define get_wch(c)     wget_wch(stdscr,c)
#define get_wstr(t)    wget_wstr(stdscr,t)
#define getbkgrnd(wch) wgetbkgrnd(stdscr,wch)
#define getn_wstr(t,n) wgetn_wstr(stdscr,t,n)
#define hline_set(c,n) whline_set(stdscr,c,n)
#define in_wch(c)      win_wch(stdscr,c)
#define in_wchstr(c)   win_wchstr(stdscr,c)
#define innwstr(c,n)   winnwstr(stdscr,c,n)
#define ins_nwstr(t,n) wins_nwstr(stdscr,t,n)
#define ins_wch(c)     wins_wch(stdscr,c)
#define ins_wstr(t)    wins_wstr(stdscr,t)
#define inwstr(c)      winwstr(stdscr,c)
#define box_set(w,v,h)  wborder_set(w,v,v,h,h,0,0,0)
#define vline_set(c,n) wvline_set(stdscr,c,n)
#define wadd_wchstr(win,str)    wadd_wchnstr(win,str,-1)
#define mvadd_wch(y,x,c)        mvwadd_wch(stdscr,y,x,c)
#define mvadd_wchstr(y,x,s)     mvwadd_wchstr(stdscr,y,x,s)
#define mvvline_set(y,x,c,n)    mvwvline_set(stdscr,y,x,c,n)
#define mvwget_wch(y,x,c)       mwwget_wch(stdscr,y,x,c)
#define mvwget_wchstr(y,x,s)    mwwget_wchstr(stdscr,y,x,s)
#define mvwgetbkgrnd(y,x,wch)   mwwgetbkgrnd(stdscr,y,x,wch)
15 Utility Libraries

LSB Core - Generic 5.0

win_wchstr(win,c))
#define mvwinnwstr(win,y,x,c,n) (wmove(win,y,x) == ERR ? ERR :
  winwstr(win,c))
#define mvwulinestset(win,y,x,c,n) (wmove(win,y,x) == ERR ? ERR :
  wline_set(win,c))
#define mvwvline_set(win,y,x,c,n)       (wmove(win,y,x) == ERR ? ERR :
  wvline_set(win,c))
#define NCURSES_MOUSE_MASK(b,m) ((m) << (((b) - 1) * 6))
#define NCURSES_BUTTON_RELEASED 001L
#define NCURSES_BUTTON_PRESSED  002L
#define NCURSES_BUTTON_CLICKED  004L
#define NCURSES_DOUBLE_CLICKED  010L
#define NCURSES_TRIPLE_CLICKED  020L
#define NCURSES_RESERVED_EVENT  040L
#define BUTTON1_RELEASED         NCURSES_MOUSE_MASK(1,
  NCURSES_BUTTON_RELEASED)
#define BUTTON1_PRESSED  NCURSES_MOUSE_MASK(1,
  NCURSES_BUTTON_PRESSED)
#define BUTTON1_CLICKED  NCURSES_MOUSE_MASK(1,
  NCURSES_BUTTON_CLICKED)
#define BUTTON1_DOUBLE_CLICKED   NCURSES_MOUSE_MASK(1,
  NCURSES_DOUBLE_CLICKED)
#define BUTTON1_TRIPLE_CLICKED   NCURSES_MOUSE_MASK(1,
  NCURSES_TRIPLE_CLICKED)
#define BUTTON2_RELEASED         NCURSES_MOUSE_MASK(2,
  NCURSES_BUTTON_RELEASED)
#define BUTTON2_PRESSED  NCURSES_MOUSE_MASK(2,
  NCURSES_BUTTON_PRESSED)
#define BUTTON2_CLICKED  NCURSES_MOUSE_MASK(2,
  NCURSES_BUTTON_CLICKED)
#define BUTTON2_DOUBLE_CLICKED   NCURSES_MOUSE_MASK(2,
  NCURSES_DOUBLE_CLICKED)
#define BUTTON2_TRIPLE_CLICKED   NCURSES_MOUSE_MASK(2,
  NCURSES_TRIPLE_CLICKED)
#define BUTTON3_RELEASED         NCURSES_MOUSE_MASK(3,
  NCURSES_BUTTON_RELEASED)
#define BUTTON3_PRESSED  NCURSES_MOUSE_MASK(3,
  NCURSES_BUTTON_PRESSED)
#define BUTTON3_CLICKED  NCURSES_MOUSE_MASK(3,
  NCURSES_BUTTON_CLICKED)
#define BUTTON3_DOUBLE_CLICKED   NCURSES_MOUSE_MASK(3,
  NCURSES_DOUBLE_CLICKED)
#define BUTTON3_TRIPLE_CLICKED   NCURSES_MOUSE_MASK(3,
  NCURSES_TRIPLE_CLICKED)
#define BUTTON4_RELEASED         NCURSES_MOUSE_MASK(4,
  NCURSES_BUTTON_RELEASED)
#define BUTTON4_PRESSED  NCURSES_MOUSE_MASK(4,
  NCURSES_BUTTON_PRESSED)
#define BUTTON4_CLICKED  NCURSES_MOUSE_MASK(4,
  NCURSES_BUTTON_CLICKED)
#define BUTTON4_DOUBLE_CLICKED   NCURSES_MOUSE_MASK(4,
  NCURSES_DOUBLE_CLICKED)
#define BUTTON4_TRIPLE_CLICKED   NCURSES_MOUSE_MASK(4,
  NCURSES_TRIPLE_CLICKED)
#define BUTTON5_RELEASED         NCURSES_MOUSE_MASK(5,
  NCURSES_BUTTON_RELEASED)
#define BUTTON5_PRESSED  NCURSES_MOUSE_MASK(5,
  NCURSES_BUTTON_PRESSED)
#define BUTTON5_CLICKED  NCURSES_MOUSE_MASK(5,
  NCURSES_BUTTON_CLICKED)
#define BUTTON5_DOUBLE_CLICKED   NCURSES_MOUSE_MASK(5, NCURSES_DOUBLE_CLICKED)
#define BUTTON5_TRIPLE_CLICKED   NCURSES_MOUSE_MASK(5, NCURSES_TRIPLE_CLICKED)
#define BUTTON_CTRL     NCURSES_MOUSE_MASK(6, 0001L)
#define BUTTON_SHIFT    NCURSES_MOUSE_MASK(6, 0002L)
#define BUTTON_ALT      NCURSES_MOUSE_MASK(6, 0004L)
#define REPORT_MOUSE_POSITION   NCURSES_MOUSE_MASK(6, 0010L)
#define BUTTON1_RESERVED_EVENT   NCURSES_MOUSE_MASK(1, NCURSES_RESERVED_EVENT)
#define BUTTON2_RESERVED_EVENT   NCURSES_MOUSE_MASK(2, NCURSES_RESERVED_EVENT)
#define BUTTON3_RESERVED_EVENT   NCURSES_MOUSE_MASK(3, NCURSES_RESERVED_EVENT)
#define BUTTON4_RESERVED_EVENT   NCURSES_MOUSE_MASK(4, NCURSES_RESERVED_EVENT)
#define ALL_MOUSE_EVENTS        (REPORT_MOUSE_POSITION - 1)
#define BUTTON_RELEASE(e,       x) ((e) & NCURSES_MOUSE_MASK(x, 001))
#define BUTTON_PRESS(e, x) ((e) & NCURSES_MOUSE_MASK(x, 002))
#define BUTTON_CLICK(e, x) ((e) & NCURSES_MOUSE_MASK(x, 004))
#define BUTTON_DOUBLE_CLICK(e, x) ((e) & NCURSES_MOUSE_MASK(x, 010))
#define BUTTON_TRIPLE_CLICK(e, x) ((e) & NCURSES_MOUSE_MASK(x, 020))
#define BUTTON_RESERVED_EVENT(e, x) ((e) & NCURSES_MOUSE_MASK(x, 040))
#define mouse_trafo(y,x,to_screen) wmouse_trafo(stdscr,y,x,to_screen)
#define _tracech_t      _tracecchar_t
#define _tracech_t2     _tracecchar_t2
#define TRACE_DISABLE   0x0000
#define TRACE_TIMES     0x0001
#define TRACE_TPUTS     0x0002
#define TRACE_UPDATE    0x0004
#define TRACE_MOVE      0x0008
#define TRACE_CHARPUT   0x0010
#define TRACE_ORDINARY  0x001F
#define TRACE_CALLS     0x0020
#define TRACE_VIRTPUT   0x0040
#define TRACE_IEVENT    0x0080
#define TRACE_BITS      0x0100
#define TRACE_ICALLS    0x0200
#define TRACE_CCALLS    0x0400
#define TRACE_DATABASE  0x0800
#define TRACE_ATTRS     0x1000
#define TRACE_SHIFT     13
#define TRACE_MAXIMUM   ((1 << TRACE_SHIFT) - 1)
#define OPTIMIZE_MVCUR  0x01
#define OPTIMIZE_HASHMAP        0x02
#define OPTIMIZE_SCROLL 0x04
#define OPTIMIZE_ALL    0xff

typedef unsigned long int chtype;
typedef chtype attr_t;

typedef struct pdat {
    short _pad_y;
    short _pad_x;
    short _pad_top;
    short _pad_left;
    short _pad_bottom;
    short _pad_right;
};
typedef struct screen SCREEN;
typedef struct _win_st WINDOW;
typedef unsigned long int mmask_t;
typedef unsigned char bool;

typedef unsigned char NCURSES_BOOL;
typedef int (*NCURSES_OUTC) (int);
typedef int (*NCURSES_WINDOW_CB) (WINDOW *, void *);
typedef int (*NCURSES_SCREEN_CB) (SCREEN *, void *);

struct _win_st {
    short _cury;                /* current cursor position */
    short _curx;
    short _maxy;                /* maximums of x and y, NOT
    window size */
    short _maxx;
    short _begy;
    short _begx;                /* window size */
    short _flags;               /* \* maximums of x and y, NOT
    attr_t _attrs;              /* current attribute for non-
    \* current background
    space character */
    chtype _bkgd;               /* \* current attribute for non-
    \* current background
    char/attribute pair */
    bool _notimeout;            /* no time out on function-key
    entry? */
    bool _clear;                /* \* no time out on function-key
    \* consider all data in the
    window invalid? */
    bool _leaveok;              /* \* OK to not reset cursor on
    exit? */
    bool _scroll;               /* \* OK to scroll this window? */
    bool _idlok;                /* \* OK to use insert/delete line?
    */
    bool _idcok;                /* \* OK to use insert/delete char?
    */
    bool _immed;                /* \* window in immed mode? (not yet
    used) */
    bool _sync;                 /* \* window in sync mode? */
    bool _use_keypad;           /* \* process function keys into
    \* KEY_ symbols? */
    int _delay;                 /* = delay */
    struct ldat *_line;         /* \* the actual line data */
    short _regtop;              /* \* top line of scrolling region
    */
    short _regbottom;           /* \* bottom line of scrolling
    region */
    int _parx;                  /* \* x coordinate of this window in
    parent */
    int _pary;                  /* \* y coordinate of this window in
    parent */
    WINDOW * _parent;           /* \* pointer to parent if a sub-
    window */
    struct pdat _pad;
    short _yoffset;             /* \* real begy is _begy + _yoffset
    */
    chtype _bkgrnd;             /* \* current background
    char/attribute pair */
};

extern int COLORS;
extern int COLOR_PAIRS;
extern int COLS;
extern int LINES;
extern chtype acs_map[];
extern int add_wch(cchar_t *);
extern int add_wchnstr(cchar_t *, int);
extern int add_wchstr(cchar_t *);
extern int addch(const chtype);
extern int addchnstr(const chtype *, int);
extern int addchstr(const chtype *);
extern int addnstr(const char *, int);
extern int addnwstr(wchar_t *, int);
extern int addstr(const char *);
extern int addwstr(wchar_t *);
extern int assume_default_colors(int, int);
extern int attr_get(attr_t *, short *, void *);
extern int attr_off(attr_t, void *);
extern int attr_on(attr_t, void *);
extern int attr_set(attr_t, short, void *);
extern int attr_t_set(const attr_t *, short, void *);
extern int attr_t_set(const attr_t *, short, void *);
extern int border(chtype, chtype, chtype, chtype, chtype, chtype, chtype, chtype);
extern int border_set(cchar_t *, cchar_t *, cchar_t *, cchar_t *, cchar_t *, cchar_t *, cchar_t *, cchar_t *);
extern int box(WINDOW *, chtype, chtype);
extern int box_set(WINDOW *, cchar_t *, cchar_t *);
extern unsigned char can_change_color(void);
extern int cbreak(void);
extern int chgat(int, attr_t, short, const void *);
extern int clear(void);
extern int clearok(WINDOW *, unsigned char);
extern int clrtoeol(void);
extern int clrtobol(void);
extern int color_content(short, short *, short *, short *);
extern int color_set(short, void *);
extern int copywin(const WINDOW *, WINDOW *, int, int, int, int, int);
extern int curs_set(int);
extern WINDOW *curscr;
extern const char *curses_version(void);
extern int def_prog_mode(void);
extern int def_shell_mode(void);
extern int define_key(const char *, int);
extern int delay_output(int);
extern int delch(void);
extern int deleteeln(void);
extern void delscreen(SCREEN *);
extern int delwin(WINDOW *);
extern WINDOW *derwin(WINDOW *, int, int, int, int);
extern int doupdate(void);
extern WINDOW *dupwin(WINDOW *);
extern int echo(void);
extern int echo_wchar(cchar_t *);
extern int echochar(const chtype);
extern int endwin(void);
extern int erase(void);
extern char erasechar(void);
extern int erasewchar(wchar_t *);
extern void filter(void);
extern int flash(void);
extern int flushinp(void);
extern int get_wch(wint_t *);
extern int get_wstr(wint_t *);
extern chtype getbkgd(WINDOW *);
extern int getbkgrnd(cchar_t *);
extern int getcchar(cchar_t *, wchar_t *, attr_t *, short *, void *);
extern int getch(void);
extern int getmouse(MEVENT *);
extern int getn_wstr(wint_t *, int);
extern int getnstr(char *, int);
extern int getstr(char *);
extern WINDOW *getwin(FILE *);
extern int halfdelay(int);
extern unsigned char has_colors(void);
extern unsigned char has_ic(void);
extern unsigned char has_il(void);
extern int has_key(void);
extern bool has_mouse(void);
extern int hline(chtype, int);
extern int hline_set(cchar_t *, int);
extern void idcok(WINDOW *, unsigned char);
extern int idlok(WINDOW *, unsigned char);
extern void immedok(WINDOW *, unsigned char);
extern int in_wch(cchar_t *);
extern int in_wchnstr(cchar_t *, int);
extern int in_wchstr(cchar_t *);
extern chtype inch(void);
extern int inchnstr(chtype *, int);
extern int inchstr(chtype *);
extern int init_color(short, short, short, short);
extern int init_pair(short, short, short);
extern WINDOW *initscr(void);
extern int innstr(char *, int);
extern int innwstr(wchar_t *, int);
extern int ins_nwstr(wchar_t *, int);
extern int ins_wch(cchar_t *);
extern int ins_wstr(wchar_t *);
extern int insch(chtype);
extern int insdelln(int);
extern int insertln(void);
extern int insnstr(const char *, int);
extern int insstr(const char *);
extern int intrflush(WINDOW *, unsigned char);
extern int inwstr(wchar_t *);
extern unsigned char is_linetouched(WINDOW *, int);
extern unsigned char is_wintouched(WINDOW *);
extern unsigned char isendwin(void);
extern char *key_name(wchar_t);
extern char *keybound(int, int);
extern char *keyname(int);
extern int keyok(int, unsigned char);
extern int keypad(WINDOW *, unsigned char);
extern char killchar(void);
extern int killwchar(wchar_t *);
extern int leaveok(WINDOW *, unsigned char);
extern char *longname(void);
extern int mcprint(void);
extern int meta(WINDOW *, unsigned char);
extern bool mouse_trafo(int *, int *, bool);
extern int mouseinterval(int);
extern mmask_t mousemask(mmask_t, mmask_t *);
extern int move(int, int);
extern int mvadd_wch(int, int, cchar_t *);
extern int mvadd_wchnstr(int, int, cchar_t *, int);
extern int mvadd_wchstr(int, int, cchar_t *);
extern int mvaddch(const chtype, const chtype, const chtype);
extern int mvaddchnstr(int, int, const chtype *, int);
extern int mvaddchstr(int, int, const chtype *);
extern int mvaddnstr(int, int, const char *, int);
extern int mvaddmstr(int, int, wchar_t *, int);
extern int mvaddstr(int, int, const char *);
extern int mvaddwstr(int, int, wchar_t *);
extern int mvchgat(int, int, int, attr_t, short, const void *);
extern int mvcur(int, int, int, int);
extern int mvdelch(int, int);
extern int mvderwin(WINDOW *, int, int);
extern int mvget_wch(int, int, wint_t *);
extern int mvget_wstr(int, int, wint_t *);
extern int mvgetch(int, int);
extern int mvgetn_wstr(int, int, wint_t *, int);
extern int mvgetnstr(int, int, char *, int);
extern int mvgetstr(int, int, char *);
extern int mvhline(int, int, chtype, int);
extern int mvhline_set(int, int, chtype, int);
extern int mvins_wch(int, int, cchar_t *);
extern int mvins_wchstr(int, int, cchar_t *, int);
extern int mvins_wchnstr(int, int, cchar_t *, int);
extern int mvinsch(int, int, chtype);
extern int mvinsnstr(int, int, const char *, int);
extern int mvinsstr(int, int, const char *);
extern int mvinsstr(int, int, char *);
extern int mvinswstr(int, int, wchar_t *);
extern int mvinnstr(int, int, const char *, ...);
extern int mvinnwstr(int, int, wchar_t *, int);
extern int mvins_nwstr(int, int, wchar_t *, int);
extern int mvins_wch(int, int, cchar_t *);
extern int mvins_wchstr(int, int, cchar_t *);
extern int mvinschnstr(int, int, const char *, int);
extern int mvwadd_wch(WINDOW *, int, int, cchar_t *);
extern int mvwadd_wchnstr(WINDOW *, int, int, cchar_t *, int);
extern int mvwadd_wchstr(WINDOW *, int, int, cchar_t *);
extern int mvwaddch(const chtype, const chtype, const chtype, const chtype);
extern int mvwaddchnstr(WINDOW *, int, int, const chtype *, int);
extern int mvwaddchstr(WINDOW *, int, int, const chtype *);
extern int mvwaddnstr(WINDOW *, int, int, const char *, int);
extern int mvwaddnwstr(WINDOW *, int, int, wchar_t *, int);
extern int mvwaddwstr(WINDOW *, int, int, wchar_t *);
extern int mvwhline(WINDOW *, int, int, chtype, int);
extern int mvwhline_set(WINDOW *, int, int, cchar_t *, int);
extern int mvwaddnstr(WINDOW *, int, int, const char *, int);
extern int mvwaddstr(WINDOW *, int, int, const char *);
extern int mvwaddwstr(WINDOW *, int, int, wchar_t *);
extern int mvwchgat(WINDOW *, int, int, attr_t, short, const void *);
extern int mvwaddch(WINDOW *, int, int);
extern int mvwaddch(WINDOW *, int, int, wint_t *);
extern int mvwaddstr(WINDOW *, int, int, wint_t *);
extern int mvwaddchnstr(WINDOW *, int, int, cchar_t *, int);
extern int mvwaddchstr(WINDOW *, int, int, cchar_t *);
extern int mvwaddnstr(WINDOW *, int, int, char *, int);
extern int mvwaddnwstr(WINDOW *, int, int, wchar_t *, int);
extern int mvwaddwstr(WINDOW *, int, int, wchar_t *);
extern int mvwaddnstr(WINDOW *, int, int, const char *, int);
extern int mvwaddstr(WINDOW *, int, int, const char *);
extern int mvwaddwstr(WINDOW *, int, int, wchar_t *);
extern int mvwhline(WINDOW *, int, int, chtype, int);
extern int mvwhline_set(WINDOW *, int, int, cchar_t *, int);
extern int mvwaddnstr(WINDOW *, int, int, const char *, int);
extern int mvwaddstr(WINDOW *, int, int, char *);
extern int mvwinnwstr(WINDOW *, int, int, wchar_t *, int);
extern int mvwins_nwstr(WINDOW *, int, int, wchar_t *, int);
extern int mvwins_wch(WINDOW *, int, int, cchar_t *);
extern int mvwins_wstr(WINDOW *, int, int, wchar_t *);
extern int mvwinsch(WINDOW *, int, int, chtype);
extern int mvwinsnstr(WINDOW *, int, int, const char *, int);
extern int mvwinsstr(WINDOW *, int, int, const char *);
extern int mvwinstr(WINDOW *, int, int, char *);
extern int mvwinwstr(WINDOW *, int, int, wchar_t *);
extern int mvwprintw(WINDOW *, int, int, const char *, ...);
extern int mvwscanw(WINDOW *, int, int, char *, ...);
extern int mvwvline(WINDOW *, int, int, chtype, int);
extern int mvwvline_set(WINDOW *, int, int, cchar_t *, int);
extern int napms(int);
extern WINDOW *newpad(int, int);
extern WINDOW *newscr;
extern SCREEN *newterm(char *, FILE *, FILE *);
extern WINDOW *newwin(int, int, int, int);
extern int nl(void);
extern int nocbreak(void);
extern int nodelay(WINDOW *, unsigned char);
extern int noecho(void);
extern int nonl(void);
extern void noqiflush(void);
extern int noraw(void);
extern int notimeout(WINDOW *, unsigned char);
extern int overlay(const WINDOW *, WINDOW *);
extern int overwrite(const WINDOW *, WINDOW *);
extern int pair_content(short, short *, short *);
extern int pecho_wchar(WINDOW *, cchar_t *);
extern int pechochar(const chtype, const chtype);
extern int pnoutrefresh(WINDOW *, int, int, int, int, int, int);
extern int printfw(const char *, ...);
extern int putwin(WINDOW *, FILE *);
extern void qiflush(void);
extern int raw(void);
extern int redrawwin(WINDOW *);
extern int refresh(void);
extern int reset_prog_mode(void);
extern int reset_shell_mode(void);
extern int resizeterm(int, int);
extern int ripoffline(int, int (*)(WINDOW *, int));
extern int savetty(void);
extern char *scr_dump(const char *);
extern int scr_init(const char *);
extern int scr_restore(const char *);
extern int scr_set(const char *);
extern int setcchar(cchar_t *, wchar_t *, attr_t, short, void *);
extern int setscrreg(int, int);
extern int slk_attr(void);
extern int slk_attr_off(const attr_t, void *);
extern int slk_attr_on(attr_t, void *);
extern int slk_attr_set(const attr_t, short, void *);
extern int slk_attroff(const chtype);
extern int slk_atrroff(const chtype);
extern int slk_attrset(const chtype);
extern int slk_clear(void);
extern int slk_color(short);
extern int slk_init(int);
extern char *slk_label(int);
extern int slk_noutrefresh(void);
extern int slk_refresh(void);
extern int slk_restore(void);
extern int slk_set(int, const char *, int);
extern int slk_touch(void);
extern int slk_wset(int, const wchar_t *, int);
extern int standend(void);
extern int standout(void);
extern int start_color(void);
extern WINDOW *stdscr;
extern WINDOW *subpad(WINDOW *, int, int, int, int);
extern WINDOW *subwin(WINDOW *, int, int, int, int);
extern int syncok(WINDOW *, unsigned char);
extern attr_t term_attrs(void);
extern chtype termattrs(void);
extern char *termname(void);
extern void timeout(int);
extern int touchline(WINDOW *, int, int);
extern int typeahead(int);
extern char *unctrl(chtype);
extern int unget_wch(wchar_t);
extern int ungetch(int);
extern int ungetmouse(MEVENT *);
extern int untouchwin(WINDOW *);
extern int use_default_colors(void);
extern void use_env(unsigned char);
extern int use_extended_names(unsigned char);
extern int vid_attr(attr_t, short, void *);
extern int vid_puts(attr_t, short, void *, int);
extern int vidattr(chtype);
extern int vidputs(chtype, NCURSES_OUTC);
extern int vline(chtype, int);
extern int vline_set(cchar_t *, int);
extern int vw_printw(WINDOW *, const char *, va_list);
extern int vw_scanw(WINDOW *, char *, va_list);
extern int vwprintw(WINDOW *, const char *, va_list);
extern int vwscanw(WINDOW *, char *, va_list);
extern int wbkgd(WINDOW *, chtype);
extern void wbkgdset(WINDOW *, chtype);
extern int wbkgrnd(WINDOW *, cchar_t *);
extern void wbkgrndset(WINDOW *, cchar_t *);
extern int wborder(WINDOW *, chtype, chtype, chtype, chtype,
chtype, chtype, chtype);
extern int wborder_set(WINDOW *, cchar_t *, cchar_t *, cchar_t *,
cchar_t *, cchar_t *, cchar_t *, cchar_t *,
cchar_t *, cchar_t *,

cchar_t *);
extern int wchgt(WINDOW *, int, attr_t, short, const void *);
extern int wclear(WINDOW *);
extern int wclrtobot(WINDOW *);
extern int wcrtomov(WINDOW *);
extern int wcolor_set(WINDOW *, short, void *);
extern void wcursyncup(WINDOW *);
extern int wdelch(WINDOW *);
extern int wdeleteln(WINDOW *);
extern int wecho_wchar(WINDOW *, cchar_t *);
extern int wechochar(const chtype, const chtype);
extern int werase(WINDOW *);
extern int wgetch(WINDOW *);
extern int wget_wch(WINDOW *, wint_t *);
extern int wget_wstr(WINDOW *, wint_t *);
extern int wgetwchstr(WINDOW *, cchar_t *);
extern int wgetchstr(WINDOW *);
extern int wgetn_wstr(WINDOW *, char *, int);
extern int wgetstr(WINDOW *, char *, int);
extern int whline(WINDOW *, chtype, int);
extern int whline_set(WINDOW *, cchar_t *, int);
extern int win_wch(WINDOW *, cchar_t *);
extern int win_wchnstr(WINDOW *, cchar_t *, int);
extern int winch(WINDOW *);
extern int winchstr(WINDOW *, chtype *);
extern int winnstr(WINDOW *, char *, int);
extern int winnstr(WINDOW *, char *, int);
extern int wins_wstr(WINDOW *, wchar_t *, int);
extern int wins_wchstr(WINDOW *, wchar_t *, int);
extern int winsch(WINDOW *, chtype);
extern int winsdelln(WINDOW *, int);
extern int winsertln(WINDOW *);
extern int winsnstr(WINDOW *, const char *, int);
extern int winsnstr(WINDOW *, const char *, int);
extern int winnstr(WINDOW *, const char *, int);
extern int winwstr(WINDOW *, wchar_t *, int);
extern int winwstr(WINDOW *, wchar_t *, int);
extern int wmove(WINDOW *, int, int);
extern int wnoutrefresh(WINDOW *);
extern int wprintw(WINDOW *, const char *, ...);
extern int wrefresh(WINDOW *);
extern int wresize(WINDOW *, int, int);
extern int wscrl(WINDOW *, int);
extern int wscur(WINDOW *, int);
extern int wscurc(WINDOW *, int);
extern void wsyncdown(WINDOW *);
extern void wsyncup(WINDOW *);
extern void wtimeout(WINDOW *, int);
extern int wtimeout(WINDOW *, int);
extern int wtofsln(WINDOW *, int, int, int);
extern wchar_t *wunctrl(cchar_t *);
extern int wvline(WINDOW *, chtype, int);
extern int wvline_set(WINDOW *, cchar_t *, int);

15.9.2 ncursesw/ncurses_dll.h

#define NCURSES_API
#define NCURSES_IMPEXP
#define NCURSES_STATIC
#define NCURSES_WRAPPED_VAR(type, name)   extern  type
NCURSES_PUBLIC_VAR(name)(void)
#define NCURSES_PUBLIC_VAR(name)        _nc_ ##name

15.9.3 ncursesw/term.h

#define NCURSES_TERM_H_incl     1
#define NCURSES_VERSION "5.9"
#define NCURSES_SBOOLE char
#define NCURSES_XNAMES 1
#define TERMIOS 1
#define TTY struct termios
#define TCSANOW TCSETA
#define TCSETADRAIN TCSETAW
#define TCSETAF
#define tcsetattr(fd, cmd, arg) ioctl(fd, cmd, arg)
#define tcgetattr(fd, arg)      ioctl(fd, TCGETA, arg)
#define cfgetospeed(t)  ((t)->c_cflag & CBAUD)
#define TCOFLUSH 1
#define TCIOFLUSH 2
#define tcflush(fd, arg)        ioctl(fd, TCFLSH, arg)
#define GET_TTY(fd, buf)        tcgetattr(fd, buf)
#define SET_TTY(fd, buf)        tcsetattr(fd, TCSADRAIN, buf)
#define NAMESIZE        256
#define CUR cur_term->type.
#define auto_left_margin        CUR Booleans[0]
#define auto_right_margin       CUR Booleans[1]
#define no_esc_ctlc     CUR Booleans[2]
#define ceol_standout_glitch    CUR Booleans[3]
#define eat_newline_glitch      CUR Booleans[4]
#define erase_overstrike        CUR Booleans[5]
#define generic_type    CUR Booleans[6]
#define hard_copy       CUR Booleans[7]
#define has_meta_key    CUR Booleans[8]
#define has_status_line CUR Booleans[9]
#define insert_null_glitch      CUR Booleans[10]
#define memory_above    CUR Booleans[11]
#define memory_below    CUR Booleans[12]
#define move_insert_mode        CUR Booleans[13]
#define move_standout_mode      CUR Booleans[14]
#define over_strike        CUR Booleans[15]
#define status_line_esc_ok      CUR Booleans[16]
#define dest_tabs_magic_smso    CUR Booleans[17]
#define tilde_glitch    CUR Booleans[18]
#define transparent_underline CUR Booleans[19]
#define xon_xoff        CUR Booleans[20]
#define needs_xon_xoff CUR Booleans[21]
#define prtr_silent     CUR Booleans[22]
#define hard_cursor     CUR Booleans[23]
#define non_rev_rmcup   CUR Booleans[24]
#define no_pad_char     CUR Booleans[25]
#define non_dest_scroll_region CUR Booleans[26]
#define can_change CUR Booleans[27]
#define back_color_erase CUR Booleans[28]
#define hue_lightness_saturation CUR Booleans[29]
#define col_addr_glitch CUR Booleans[30]
#define cr_cancels_micro_mode CUR Booleans[31]
#define has_print_wheel CUR Booleans[32]
#define row_addr_glitch CUR Booleans[33]
#define semi_auto_right_margin CUR Booleans[34]
#define cpi_changes_res CUR Booleans[35]
#define lpi_changes_res CUR Booleans[36]
#define columns CUR Numbers[0]
#define init_tabs CUR Numbers[1]
#define lines CUR Numbers[2]

© 2015 Linux Foundation
```c
#define lines_of_memory     CUR Numbers[3]
#define magic_cookie_glitch CUR Numbers[4]
#define padding_baud_rate   CUR Numbers[5]
#define virtual_terminal    CUR Numbers[6]
#define width_status_line   CUR Numbers[7]
#define num_labels          CUR Numbers[8]
#define label_height        CUR Numbers[9]
#define label_width         CUR Numbers[10]
#define max_attributes      CUR Numbers[11]
#define maximum_windows     CUR Numbers[12]
#define max_colors          CUR Numbers[13]
#define max_pairs           CUR Numbers[14]
#define no_color_video      CUR Numbers[15]
#define buffer_capacity     CUR Numbers[16]
#define dot_vert_spacing    CUR Numbers[17]
#define dot_horz_spacing    CUR Numbers[18]
#define max_micro_jump      CUR Numbers[20]
#define micro_col_size      CUR Numbers[21]
#define micro_line_size     CUR Numbers[22]
#define number_of_pins      CUR Numbers[23]
#define output_res_char     CUR Numbers[24]
#define output_res_line     CUR Numbers[25]
#define output_res_horz_inch CUR Numbers[26]
#define output_res_vert_inch CUR Numbers[27]
#define print_rate          CUR Numbers[28]
#define wide_char_size      CUR Numbers[29]
#define buttons             CUR Numbers[30]
#define bit_image_entwining CUR Numbers[31]
#define bit_image_type      CUR Numbers[32]
#define back_tab            CUR Strings[0]
#define bell                CUR Strings[1]
#define carriage_return     CUR Strings[2]
#define change_scroll_region CUR Strings[3]
#define clear_all_tabs      CUR Strings[4]
#define clear_screen        CUR Strings[5]
#define clr_eol             CUR Strings[6]
#define clr_eos             CUR Strings[7]
#define column_address      CUR Strings[8]
#define command_character   CUR Strings[9]
#define cursor_address      CUR Strings[10]
#define cursor_down         CUR Strings[11]
#define cursor_home         CUR Strings[12]
#define cursor_invisible    CUR Strings[13]
#define cursor_left         CUR Strings[14]
#define cursor_mem_address  CUR Strings[15]
#define cursor_normal       CUR Strings[16]
#define cursor_right        CUR Strings[17]
#define cursor_to_ll        CUR Strings[18]
#define cursor_up           CUR Strings[19]
#define cursor_visible      CUR Strings[20]
#define delete_character    CUR Strings[21]
#define dis_status_line     CUR Strings[22]
#define down_half_line      CUR Strings[23]
#define enter_alt_charset_mode CUR Strings[25]
#define enter_blink_mode    CUR Strings[26]
#define enter_bold_charset_mode CUR Strings[27]
#define enter_ca_mode       CUR Strings[28]
#define enter_delete_mode   CUR Strings[29]
#define enter_dim_mode      CUR Strings[30]
#define enter_insert_mode   CUR Strings[31]
#define enter_secure_mode   CUR Strings[32]
#define enter_protected_mode CUR Strings[33]
#define enter_reverse_mode  CUR Strings[34]
#define enter_standout_mode CUR Strings[35]
```
```c
#define enter_underline_mode  CUR Strings[36]
#define erase_chars     CUR Strings[37]
#define exit_alt_charset_mode   CUR Strings[38]
#define exit_attribute_mode     CUR Strings[39]
#define exit_ca_mode    CUR Strings[40]
#define exit_delete_mode        CUR Strings[41]
#define exit_insert_mode        CUR Strings[42]
#define exit_standout_mode      CUR Strings[43]
#define exit_underline_mode     CUR Strings[44]
#define flash_screen    CUR Strings[45]
#define form_feed       CUR Strings[46]
#define from_status_line        CUR Strings[47]
#define init_1string    CUR Strings[48]
#define init_2string    CUR Strings[49]
#define init_3string    CUR Strings[50]
#define init_file       CUR Strings[51]
#define insert_character        CUR Strings[52]
#define insert_line     CUR Strings[53]
#define insert_padding  CUR Strings[54]
#define key_backspace   CUR Strings[55]
#define key_catab       CUR Strings[56]
#define key_clear       CUR Strings[57]
#define key_ctab        CUR Strings[58]
#define key_dc  CUR Strings[59]
#define key_d1 CUR Strings[60]
#define key_down        CUR Strings[61]
#define key_eic CUR Strings[62]
#define key_eol CUR Strings[63]
#define key_eos CUR Strings[64]
#define key_f0 CUR Strings[65]
#define key_f1 CUR Strings[66]
#define key_f10 CUR Strings[67]
#define key_f2 CUR Strings[68]
#define key_f3 CUR Strings[69]
#define key_f4 CUR Strings[70]
#define key_f5 CUR Strings[71]
#define key_f6 CUR Strings[72]
#define key_f7 CUR Strings[73]
#define key_f8 CUR Strings[74]
#define key_f9 CUR Strings[75]
#define key_home CUR Strings[76]
#define key_ic CUR Strings[77]
#define key_ii CUR Strings[78]
#define key_left CUR Strings[79]
#define key_ll CUR Strings[80]
#define key_npage CUR Strings[81]
#define key_ppage CUR Strings[82]
#define key_right CUR Strings[83]
#define key_sf CUR Strings[84]
#define key_sr CUR Strings[85]
#define key_stab CUR Strings[86]
#define key_up CUR Strings[87]
#define keypad_local CUR Strings[88]
#define keypad_xmit CUR Strings[89]
#define lab_f0 CUR Strings[90]
#define lab_f1 CUR Strings[91]
#define lab_f10 CUR Strings[92]
#define lab_f2 CUR Strings[93]
#define lab_f3 CUR Strings[94]
#define lab_f4 CUR Strings[95]
#define lab_f5 CUR Strings[96]
#define lab_f6 CUR Strings[97]
#define lab_f7 CUR Strings[98]
#define lab_f8 CUR Strings[99]
#define lab_f9 CUR Strings[100]
#define meta_off CUR Strings[101]
```
#define meta_on CUR Strings[102]
#define newline CUR Strings[103]
#define pad_char CUR Strings[104]
#define parm_dch CUR Strings[105]
#define parm_delete_line CUR Strings[106]
#define parm_down_cursor CUR Strings[107]
#define parm_ich CUR Strings[108]
#define parm_index CUR Strings[109]
#define parm_insert_line CUR Strings[110]
#define parm_left_cursor CUR Strings[111]
#define parm_right_cursor CUR Strings[112]
#define parm_rindex CUR Strings[113]
#define parm_up_cursor CUR Strings[114]
#define pkey_key CUR Strings[115]
#define pkey_local CUR Strings[116]
#define pkey_xmit CUR Strings[117]
#define print_screen CUR Strings[118]
#define prtr_off CUR Strings[119]
#define prtr_on CUR Strings[120]
#define repeat_char CUR Strings[121]
#define reset_1string CUR Strings[122]
#define reset_2string CUR Strings[123]
#define reset_3string CUR Strings[124]
#define reset_file CUR Strings[125]
#define restore_cursor CUR Strings[126]
#define row_address CUR Strings[127]
#define save_cursor CUR Strings[128]
#define scroll_forward CUR Strings[129]
#define scroll_reverse CUR Strings[130]
#define set_attributes CUR Strings[131]
#define set_tab CUR Strings[132]
#define set_window CUR Strings[133]
#define tab CUR Strings[134]
#define to_status_line CUR Strings[135]
#define underline_char CUR Strings[136]
#define up_half_line CUR Strings[137]
#define init_prog CUR Strings[138]
#define key_a1 CUR Strings[139]
#define key_a3 CUR Strings[140]
#define key_b2 CUR Strings[141]
#define key_c1 CUR Strings[142]
#define key_c3 CUR Strings[143]
#define prtr_non CUR Strings[144]
#define char_padding CUR Strings[145]
#define acs_chars CUR Strings[146]
#define plab_norm CUR Strings[147]
#define key_btab CUR Strings[148]
#define enter_xon_mode CUR Strings[149]
#define exit_xon_mode CUR Strings[150]
#define enter_am_mode CUR Strings[151]
#define exit_am_mode CUR Strings[152]
#define xon_character CUR Strings[153]
#define xoff_character CUR Strings[154]
#define ena_acs CUR Strings[155]
#define label_on CUR Strings[156]
#define label_off CUR Strings[157]
#define key_beg CUR Strings[158]
#define key_cancel CUR Strings[159]
#define key_close CUR Strings[160]
#define key_command CUR Strings[161]
#define key_copy CUR Strings[162]
#define key_create CUR Strings[163]
#define key_end CUR Strings[164]
#define key_enter CUR Strings[165]
#define key_exit CUR Strings[166]
#define key_find CUR Strings[167]
#define key_help CUR Strings[168]
#define key_mark CUR Strings[169]
#define key_message CUR Strings[170]
#define key_move CUR Strings[171]
#define key_next CUR Strings[172]
#define key_open CUR Strings[173]
#define key_options CUR Strings[174]
#define key_previous CUR Strings[175]
#define key_print CUR Strings[176]
#define key_undo CUR Strings[177]
#define key_reference CUR Strings[178]
#define key_refresh CUR Strings[179]
#define key_replace CUR Strings[180]
#define key_restart CUR Strings[181]
#define key_resume CUR Strings[182]
#define key_save CUR Strings[183]
#define key_suspend CUR Strings[184]
#define key_open CUR Strings[185]
#define key_sbeg CUR Strings[186]
#define key_scancel CUR Strings[187]
#define key_scommand CUR Strings[188]
#define key_scopy CUR Strings[189]
#define key_screate CUR Strings[190]
#define key_sdc CUR Strings[191]
#define key sdl CUR Strings[192]
#define key_select CUR Strings[193]
#define key_send CUR Strings[194]
#define key_seol CUR Strings[195]
#define key_sexit CUR Strings[196]
#define key_sfind CUR Strings[197]
#define key_shelp CUR Strings[198]
#define key_shome CUR Strings[199]
#define key_sic CUR Strings[200]
#define key_sleft CUR Strings[201]
#define key_smess CUR Strings[202]
#define key_smove CUR Strings[203]
#define key_snex CUR Strings[204]
#define key_sopt CUR Strings[205]
#define key_spre CUR Strings[206]
#define key_sprint CUR Strings[207]
#define key_sredo CUR Strings[208]
#define key_sreplace CUR Strings[209]
#define key_sright CUR Strings[210]
#define key_srs CUR Strings[211]
#define key_ssave CUR Strings[212]
#define key_sss CUR Strings[213]
#define key_sundo CUR Strings[214]
#define req_for_input CUR Strings[215]
#define key_f11 CUR Strings[216]
#define key_f12 CUR Strings[217]
#define key_f13 CUR Strings[218]
#define key_f14 CUR Strings[219]
#define key_f15 CUR Strings[220]
#define key_f16 CUR Strings[221]
#define key_f17 CUR Strings[222]
#define key_f18 CUR Strings[223]
#define key_f19 CUR Strings[224]
#define key_f20 CUR Strings[225]
#define key_f21 CUR Strings[226]
#define key_f22 CUR Strings[227]
#define key_f23 CUR Strings[228]
#define key_f24 CUR Strings[229]
#define key_f25 CUR Strings[230]
#define key_f26 CUR Strings[231]
#define key_f27 CUR Strings[232]
#define key_f28 CUR Strings[233]
#define key_f29 CUR Strings[234]
#define key_f30 CUR Strings[235]
#define key_f31 CUR Strings[236]
#define key_f32 CUR Strings[237]
#define key_f33 CUR Strings[238]
#define key_f34 CUR Strings[239]
#define key_f35 CUR Strings[240]
#define key_f36 CUR Strings[241]
#define key_f37 CUR Strings[242]
#define key_f38 CUR Strings[243]
#define key_f39 CUR Strings[244]
#define key_f40 CUR Strings[245]
#define key_f41 CUR Strings[246]
#define key_f42 CUR Strings[247]
#define key_f43 CUR Strings[248]
#define key_f44 CUR Strings[249]
#define key_f45 CUR Strings[250]
#define key_f46 CUR Strings[251]
#define key_f47 CUR Strings[252]
#define key_f48 CUR Strings[253]
#define key_f49 CUR Strings[254]
#define key_f50 CUR Strings[255]
#define key_f51 CUR Strings[256]
#define key_f52 CUR Strings[257]
#define key_f53 CUR Strings[258]
#define key_f54 CUR Strings[259]
#define key_f55 CUR Strings[260]
#define key_f56 CUR Strings[261]
#define key_f57 CUR Strings[262]
#define key_f58 CUR Strings[263]
#define key_f59 CUR Strings[264]
#define key_f60 CUR Strings[265]
#define key_f61 CUR Strings[266]
#define key_f62 CUR Strings[267]
#define key_f63 CUR Strings[268]
#define clr_bol CUR Strings[269]
#define clearMargins CUR Strings[270]
#define set_left_margin CUR Strings[271]
#define set_right_margin CUR Strings[272]
#define label_format CUR Strings[273]
#define set_clock CUR Strings[274]
#define display_clock CUR Strings[275]
#define remove_clock CUR Strings[276]
#define create_window CUR Strings[277]
#define goto_window CUR Strings[278]
#define hangup CUR Strings[279]
#define dial_phone CUR Strings[280]
#define quick_dial CUR Strings[281]
#define tone CUR Strings[282]
#define pulse CUR Strings[283]
#define flash_hook CUR Strings[284]
#define fixed_pause CUR Strings[285]
#define wait_tone CUR Strings[286]
#define user0 CUR Strings[287]
#define user1 CUR Strings[288]
#define user2 CUR Strings[289]
#define user3 CUR Strings[290]
#define user4 CUR Strings[291]
#define user5 CUR Strings[292]
#define user6 CUR Strings[293]
#define user7 CUR Strings[294]
#define user8 CUR Strings[295]
#define user9 CUR Strings[296]
#define orig_pair CUR Strings[297]
#define orig_colors CUR Strings[298]
#define initialize_color CUR Strings[299]
#define initialize_pair CUR Strings[300]
#define set_color_pair CUR Strings[301]
#define set_foreground CUR Strings[302]
#define set_background CUR Strings[303]
#define change_char_pitch CUR Strings[304]
#define change_line_pitch CUR Strings[305]
#define change_res_horz CUR Strings[306]
#define change_res_vert CUR Strings[307]
#define define_char CUR Strings[308]
#define enter_doublewide_mode CUR Strings[309]
#define enter_draft_quality CUR Strings[310]
#define enter_italics_mode CUR Strings[311]
#define enter_leftward_mode CUR Strings[312]
#define enter_micro_mode CUR Strings[313]
#define enter_near_letter_quality CUR Strings[314]
#define enter_normal_quality CUR Strings[315]
#define enter_shadow_mode CUR Strings[316]
#define enter_subscript_mode CUR Strings[317]
#define enter_superscript_mode CUR Strings[318]
#define enter_upward_mode CUR Strings[319]
#define exit_doublewide_mode CUR Strings[320]
#define exit_italics_mode CUR Strings[321]
#define exit_leftward_mode CUR Strings[322]
#define exit_micro_mode CUR Strings[323]
#define exit_shadow_mode CUR Strings[324]
#define exit_subscript_mode CUR Strings[325]
#define exit_superscript_mode CUR Strings[326]
#define exit_upward_mode CUR Strings[327]
#define micro_column_address CUR Strings[328]
#define micro_down CUR Strings[329]
#define micro_left CUR Strings[330]
#define micro_right CUR Strings[331]
#define micro_row_address CUR Strings[332]
#define micro_up CUR Strings[333]
#define order_of_pins CUR Strings[334]
#define parm_down_micro CUR Strings[335]
#define parm_left_micro CUR Strings[336]
#define parm_right_micro CUR Strings[337]
#define parm_up_micro CUR Strings[338]
#define select_char_set CUR Strings[339]
#define set_bottom_margin CUR Strings[340]
#define set_bottom_margin_parm CUR Strings[341]
#define set_left_margin_parm CUR Strings[342]
#define set_right_margin_parm CUR Strings[343]
#define set_top_margin CUR Strings[344]
#define set_top_margin_parm CUR Strings[345]
#define start_bit_image CUR Strings[346]
#define start_char_set_def CUR Strings[347]
#define stop_bit_image CUR Strings[348]
#define stop_char_set_def CUR Strings[349]
#define subscript_characters CUR Strings[350]
#define superscript_characters CUR Strings[351]
#define these_cause_cr CUR Strings[352]
#define zero_motion CUR Strings[353]
#define char_set_names CUR Strings[354]
#define key_mouse CUR Strings[355]
#define mouse_info CUR Strings[356]
#define req_mouse_pos CUR Strings[357]
#define get_mouse CUR Strings[358]
#define set_a_foreground CUR Strings[359]
#define set_a_background CUR Strings[360]
#define pkey_plab CUR Strings[361]
#define device_type CUR Strings[362]
#define code_set_init CUR Strings[363]
#define set0_des_seq CUR Strings[364]
#define set1_des_seq CUR Strings[365]
#define set2_des_seq CUR Strings[366]
#define set3_des_seq CUR Strings[367]
#define set_lr_margin CUR Strings[368]
#define set_tb_margin CUR Strings[369]
#define bit_image_repeat CUR Strings[370]
#define bit_image_newline CUR Strings[371]
#define bit_image_carriage_return CUR Strings[372]
#define color_names CUR Strings[373]
#define define_bit_image_region CUR Strings[374]
#define end_bit_image_region CUR Strings[375]
#define set_color_band CUR Strings[376]
#define set_page_length CUR Strings[377]
#define display_pc_char CUR Strings[378]
#define exit_pc_charset_mode CUR Strings[379]
#define enter_scancode_mode CUR Strings[380]
#define exit_scancode_mode CUR Strings[381]
#define pc_term_options CUR Strings[382]
#define scancode_escape CUR Strings[383]
#define scancode_escape CUR Strings[384]
#define alt_scancode_esc CUR Strings[385]
#define enter_horizontal_hl_mode CUR Strings[386]
#define enter_left_hl_mode CUR Strings[387]
#define enter_low_hl_mode CUR Strings[388]
#define enter_right_hl_mode CUR Strings[389]
#define enter_top_hl_mode CUR Strings[390]
#define enter_vertical_hl_mode CUR Strings[391]
#define set_a_attributes CUR Strings[392]
#define set_pglen_inch CUR Strings[393]
#define BOOLWRITE 37
#define NUMWRITE 33
#define STRWRITE 394
#define beehive_glitch no_esc_ctlc
#define teleray_glitch dest_tabs_magic_smso
#define micro_char_size micro_col_size
#define termcap_init2 CUR Strings[394]
#define termcap_reset CUR Strings[395]
#define magic_cookie_glitch_ul CUR Numbers[33]
#define backspaces_with_bs CUR Booleans[37]
#define crt_no_scrolling CUR Booleans[38]
#define no_correctly_working_cr CUR Booleans[39]
#define carriage_return_delay CUR Numbers[34]
#define new_line_delay CUR Numbers[35]
#define linefeed_if_not_lf CUR Strings[396]
#define backspace_if_not_bs CUR Strings[397]
#define gnu_has_meta_key CUR Booleans[40]
#define linefeed_is_newline CUR Booleans[41]
#define backspace_delay CUR Numbers[36]
#define horizontal_tab_delay CUR Numbers[37]
#define number_of_function_keys CUR Numbers[38]
#define other_non_function_keys CUR Strings[398]
#define arrow_key_map CUR Strings[399]
#define has_hardware_tabs CUR Booleans[42]
#define return_does_clr_eol CUR Booleans[43]
#define acs_ulcorner CUR Strings[400]
#define acs_llcorner CUR Strings[401]
#define acs_urcorner CUR Strings[402]
#define acs_lrcorner CUR Strings[403]
#define acs_ttee CUR Strings[404]
#define acs_rtee CUR Strings[405]
#define acs_hline CUR Strings[406]
#define acs_vline CUR Strings[407]
#define acs_plus CUR Strings[410]
#define memory_lock CUR Strings[411]
#define memory_unlock CUR Strings[412]
typedef struct termtype {
    char *term_names;
    char *str_table;
    char *Booleans;
    short *Numbers;
    char **Strings;
    char *ext_str_table;
    char **ext_Names;
    unsigned short num_Booleans;
    unsigned short num_Numbers;
    unsigned short num_Strings;
    unsigned short ext_Booleans;
    unsigned short ext_Numbers;
    unsigned short ext_Strings;
} TERMTYPE;

typedef struct term {
    TERMTYPE type;
    short Filedes;
    struct termios Ottyb;
    struct termios Nttyb;
    int _baudrate;
    char * _termmame;
} TERMINAL;

extern TERMINAL *cur_term;
extern int del_curterm(TERMINAL *);
extern int putp(const char *);
extern int restartterm(char *, int, int *);
extern TERMINAL *set_curterm(TERMINAL *);
extern int setupterm(char *, int, int *);
extern int tgetent(char *, const char *);
extern int tgetflag(char *);
extern int tgetnum(char *);
extern char *tgetstr(char *, char *);
extern char *tgoto(const char *, int, int);
extern int tigetflag(char *);
extern int tigetnum(char *);
extern char *tigetstr(char *);
extern char *tparm(char *, ...);
extern int tputs(const char *, int, int (*)(int));
extern char ttytype[];

15.9.4 ncursesw/unctrl.h

#define NCURSES_UNCTRL_H_incl 1
#define NCURSES_VERSION "5.9"
15.10 Interface Definitions for libncursesw

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

Other interfaces listed in Section 15.8 shall behave as described in the referenced base document.

15.11 Interfaces for libutil

Table 15-11 defines the library name and shared object name for the libutil library

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Library:</strong></td>
<td>libutil</td>
</tr>
<tr>
<td><strong>SONAME:</strong></td>
<td>libutil.so.1</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications: [LSB] This Specification

15.11.1 Utility Functions

15.11.1.1 Interfaces for Utility Functions

An LSB conforming implementation shall provide the generic functions for Utility Functions specified in Table 15-12, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th></th>
<th>[LSB]</th>
<th></th>
<th>[LSB]</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>forkpty</td>
<td>login</td>
<td>login_tty</td>
<td>logout</td>
<td></td>
</tr>
<tr>
<td>openpty</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

15.12 Data Definitions for libutil

This section defines global identifiers and their values that are associated with interfaces contained in libutil. 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.

15.12.1 pty.h

extern int forkpty(int *__amaster, char *__name, const struct termios *__termpt, const struct winsize *__winps);

extern int openpty(int *__amaster, int *__slave, char *__name, const struct termios *__termpt, const struct winsize *__winps);
15.13 Interface Definitions for libutil

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

Other interfaces listed in Section 15.11 shall behave as described in the referenced base
document.

forkpty

Name
forkpty — Create a new process attached to an available pseudo-terminal

Synopsis
#include <pty.h>
int forkpty(int * amaster, char * name, const struct termios * termp,
const struct winsize * winp);

Description
The forkpty() function shall find and open a pseudo-terminal device pair in the same
manner as the openpty() function. If a pseudo-terminal is available, forkpty() shall
create a new process in the same manner as the fork() function, and prepares the new
process for login in the same manner as login_tty().

If termp is not null, it shall refer to a termios structure that shall be used to initialize
the characteristics of the slave device. If winp is not null, it shall refer to a winsize
structure used to initialize the window size of the slave device.

Return Value
On success, the parent process shall return the process id of the child, and the child shall
return 0. On error, no new process shall be created, -1 shall be returned, and errno shall
be set appropriately. On success, the parent process shall receive the file descriptor of
the master side of the pseudo-terminal in the location referenced by amaster, and, if
name is not NULL, the filename of the slave device in name.

Errors
EAGAIN
   Unable to create a new process.

ENOENT
   There are no available pseudo-terminals.

ENOMEM
   Insufficient memory was available.
login

Name
login — login utility function

Synopsis
#include <utmp.h>
void login (struct utmp * ut);

Description
The login() function shall update the user accounting databases. The ut parameter
shall reference a utmp structure for all fields except the following:

1. The ut_type field shall be set to USER_PROCESS.
2. The ut_pid field shall be set to the process identifier for the current process.
3. The ut_line field shall be set to the name of the controlling terminal device. The
   name shall be found by examining the device associated with the standard input,
   output and error streams in sequence, until one associated with a terminal device
   is found. If none of these streams refers to a terminal device, the ut_line field
   shall be set to "???". If the terminal device is in the /dev directory hierarchy, the
   ut_line field shall not contain the leading "/dev/", otherwise it shall be set to
   the final component of the pathname of the device. If the user accounting database
   imposes a limit on the size of the ut_line field, it shall truncate the name, but
   any such limit shall not be smaller than UT_LINESIZE (including a terminating
   null character).

Return Value
None

Errors
None
login_tty

Name
login_tty — Prepare a terminal for login

Synopsis
#include <utmp.h>
int login_tty (int fdr);

Description
The login_tty() function shall prepare the terminal device referenced by the file descriptor fdr. This function shall create a new session, make the terminal the controlling terminal for the current process, and set the standard input, output, and error streams of the current process to the terminal. If fdr is not the standard input, output or error stream, then login_tty() shall close fdr.

Return Value
On success, login_tty() shall return zero; otherwise -1 is returned, and errno shall be set appropriately.

Errors
ENOTTY
fdr does not refer to a terminal device.

logout

Name
logout — logout utility function

Synopsis
#include <utmp.h>
int logout (const char * line);

Description
Given the device line, the logout() function shall search the user accounting database which is read by getutent() for an entry with the corresponding line, and with the type of USER_PROCESS. If a corresponding entry is located, it shall be updated as follows:

1. The ut_name field shall be set to zeroes (UT_NAMESIZE NUL bytes).
2. The ut_host field shall be set to zeroes (UT_HOSTSIZE NUL bytes).
3. The ut_tv shall be set to the current time of day.
4. The ut_type field shall be set to DEAD_PROCESS.

Return Value
On success, the logout() function shall return non-zero. Zero is returned if there was no entry to remove, or if the utmp file could not be opened or updated.
logwtmp

Name

logwtmp — append an entry to the wtmp file

Synopsis

#include <utmp.h>
void logwtmp (const char * line, const char * name, const char * host);

Description

If the process has permission to update the user accounting databases, the logwtmp() function shall append a record to the user accounting database that records all logins and logouts. The record to be appended shall be constructed as follows:

1. The ut_line field shall be initialized from line. If the user accounting database imposes a limit on the size of the ut_line field, it shall truncate the value, but any such limit shall not be smaller than UT_LINESIZE (including a terminating null character).

2. The ut_name field shall be initialized from name. If the user accounting database imposes a limit on the size of the ut_name field, it shall truncate the value, but any such limit shall not be smaller than UT_NAMESIZE (including a terminating null character).

3. The ut_host field shall be initialized from host. If the user accounting database imposes a limit on the size of the ut_host field, it shall truncate the value, but any such limit shall not be smaller than UT_HOSTSIZE (including a terminating null character).

4. If the name parameter does not refer to an empty string (i.e. ""), the ut_type field shall be set to USER_PROCESS; otherwise the ut_type field shall be set to DEAD_PROCESS.

5. The ut_id field shall be set to the process identifier for the current process.

6. The ut_tv field shall be set to the current time of day.

Note: If a process does not have write access to the the user accounting database, the logwtmp() function will not update it. Since the function does not return any value, an application has no way of knowing whether it succeeded or failed.

Return Value

None.
openpty

Name

openpty — find and open an available pseudo-terminal

Synopsis

#include <pty.h>
int openpty(int *amaster, int *slave, char *name, const struct termios *termp, const struct winsize *winp);

Description

The openpty() function shall find an available pseudo-terminal and return file descriptors for the master and slave devices in the locations referenced by amaster and aslave respectively. If name is not NULL, the filename of the slave shall be placed in the user supplied buffer referenced by name. If termp is not NULL, it shall point to a termios structure used to initialize the terminal parameters of the slave pseudo-terminal device. If winp is not NULL, it shall point to a winsize structure used to initialize the window size parameters of the slave pseudo-terminal device.

Return Value

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

Errors

ENOENT

There are no available pseudo-terminals.
V C++ Libraries
16 Libraries

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

16.1 Interfaces for libstdc++

Table 16-1 defines the library name and shared object name for the libstdc++ library

<table>
<thead>
<tr>
<th>Library</th>
<th>libstdc++</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME</td>
<td>libstdc++.so.6</td>
</tr>
</tbody>
</table>

Unless stated otherwise, all symbols are in the std:: namespace.

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

- [CXXABI-1.86] Itanium™ C++ ABI
- [LSB] This Specification

16.1.1 C++ Runtime Support

16.1.1.1 Interfaces for C++ Runtime Support

An LSB conforming implementation shall provide the generic methods for C++ Runtime Support specified in Table 16-2, with the full mandatory functionality as described in the referenced underlying specification.

```
<table>
<thead>
<tr>
<th>Function</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>__gnu_cxx::__atomic_add(int volatile*, int)(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>__gnu_cxx::__exchange_and_add(int volatile*, int)(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>__gnu_cxx::__verbose_terminate_handler()</td>
<td>[CXXABI-1.3]</td>
</tr>
<tr>
<td>unexpected()</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>set_terminate(void (*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>set_unexpected(void (*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>set_new_handler(void (*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_bad_cast()</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_bad_alloc()</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_bad_typeid()</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>uncaught_exception()</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_ios_failure(char const*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_logic_error(char const*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_range_error(char const*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_domain_error(char const*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_length_error(char const*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_out_of_range(char const*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_bad_exception()</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_runtime_error(char const*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_overflow_error(char const*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__throw_underflow_error(char const*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>
```
An LSB conforming implementation shall provide the generic data interfaces for C++ Runtime Support specified in Table 16-3, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-3 libstdcxx - C++ Runtime Support Data Interfaces**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cin</code></td>
<td>Input stream</td>
</tr>
<tr>
<td><code>cerr</code></td>
<td>Error stream</td>
</tr>
<tr>
<td><code>clog</code></td>
<td>Error stream</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
### 16.1.2 C++ type descriptors for built-in types

#### 16.1.2.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for C++ type descriptors for built-in types specified in Table 16-4, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Type Description</th>
<th>Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signed char const*</td>
<td>[CXXABI-1.3] [CXXABI-1.86]</td>
</tr>
<tr>
<td>Bool const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Char const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Double const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Long double const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Float const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Unsigned char const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Int const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Unsigned int const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Long const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Unsigned long const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Short const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Unsigned short const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Void const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Wchar_t const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Long long const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Unsigned long long const*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Signed char*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Bool*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Char*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Double*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Long double*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Float*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Unsigned char*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Int*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Unsigned int*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Long*</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Type Information</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>typeinfo for unsigned long*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for short*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for unsigned short*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for void*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for wchar_t*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for long long*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for unsigned long long*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for signed char(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for bool(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for char(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for double(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for long double(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for float(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for unsigned char(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for int(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for unsigned int(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for long(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for unsigned long(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for short(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for unsigned short(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for void(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for wchar_t(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for long long(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo for unsigned long long(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for signed char const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for bool const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for char const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for double const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for long double const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for float const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for unsigned char const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for int const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for unsigned int const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for long const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for unsigned long const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for short const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for unsigned short const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for void const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for wchar_t const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for long long const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for unsigned long long const*(CXXABI_1.3)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16 Libraries

LSB Core - Generic 5.0
16.1.3 C++ _Rb_tree

16.1.3.1 Interfaces for C++ _Rb_tree

An LSB conforming implementation shall provide the generic methods for C++ _Rb_tree specified in Table 16-5, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-5 libstdcxx - C++ _Rb_tree Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>_Rb_tree_decrement(_Rb_tree_node_base const*)</td>
<td>decrements the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_decrement(_Rb_tree_node_base*)</td>
<td>decrements the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_increment(_Rb_tree_node_base const*)</td>
<td>increments the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_increment(_Rb_tree_node_base*)</td>
<td>increments the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_decrement(_Rb_tree_node_base const*)</td>
<td>decrements the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_increment(_Rb_tree_node_base const*)</td>
<td>increments the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_decrement(_Rb_tree_node_base*)</td>
<td>decrements the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_increment(_Rb_tree_node_base*)</td>
<td>increments the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_decrement(_Rb_tree_node_base const*)</td>
<td>decrements the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_increment(_Rb_tree_node_base const*)</td>
<td>increments the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_decrement(_Rb_tree_node_base*)</td>
<td>decrements the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_increment(_Rb_tree_node_base*)</td>
<td>increments the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_decrement(_Rb_tree_node_base const*)</td>
<td>decrements the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_increment(_Rb_tree_node_base const*)</td>
<td>increments the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_decrement(_Rb_tree_node_base*)</td>
<td>decrements the node</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>_Rb_tree_increment(_Rb_tree_node_base*)</td>
<td>increments the node</td>
<td>GLIBCXX_3.4</td>
</tr>
</tbody>
</table>
16 Libraries

16.1.4 Class type_info

16.1.4.1 Class data for type_info

The virtual table for the std::type_info class is described by Table 16-6.

**Table 16-6 Primary vtable for type_info**

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]:</th>
<th>vfunc[1]:</th>
<th>vfunc[2]:</th>
<th>vfunc[3]:</th>
<th>vfunc[4]:</th>
<th>vfunc[5]:</th>
</tr>
</thead>
</table>
| 0           |                     | typeinfo for type_info | type_info::~type_info() | type_info::~type_info() | type_info::__is_pointer_p() const | type_info::__is_function_p() const | type_info::__do_catch(type_info const*, void**, unsigned int) const | type_info::__do_upcast(__cxxabiv1::__c

The Run Time Type Information for the std::type_info class is described by Table 16-7.

**Table 16-7 typeinfo for type_info**

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>typeinfo name for type_info</td>
</tr>
</tbody>
</table>

16.1.4.2 Interfaces for Class type_info

An LSB conforming implementation shall provide the generic methods for Class std::type_info specified in Table 16-8, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-8 libstdcxx - Class type_info Function Interfaces**

| type_info::__do_catch(type_info const*, void**, unsigned int) const(GLIBCXX_3.4) | ISOCEXX |
| type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const(GLIBCXX_3.4) | ISOCEXX |
| type_info::__is_pointer_p() const(GLIBCXX_3.4) | ISOCEXX |
| type_info::__is_function_p() const(GLIBCXX_3.4) | ISOCEXX |
| type_info::type_info(GLIBCXX_3.4) | ISOCEXX |
| type_info::~type_info(GLIBCXX_3.4) | ISOCEXX |
An LSB conforming implementation shall provide the generic data interfaces for Class std::type_info specified in Table 16-9, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-9 libstdcxx - Class type_info Data Interfaces

<table>
<thead>
<tr>
<th>Type Info</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for type_info(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for type_info(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for type_info(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.5 Class __cxxabiv1::__enum_type_info

16.1.5.1 Class data for __cxxabiv1::__enum_type_info

The virtual table for the __cxxabiv1::__enum_type_info class is described by Table 16-10.

Table 16-10 Primary vtable for __cxxabiv1::__enum_type_info

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __cxxabiv1::__enum_type_info</td>
</tr>
<tr>
<td>vfunc[0]</td>
<td>__cxxabiv1::__enum_type_info::~_enum_type_info()</td>
</tr>
<tr>
<td>vfunc[1]</td>
<td>__cxxabiv1::__enum_type_info::~_enum_type_info()</td>
</tr>
<tr>
<td>vfunc[2]</td>
<td>type_info::__is_pointer_p() const</td>
</tr>
<tr>
<td>vfunc[3]</td>
<td>type_info::__is_function_p() const</td>
</tr>
<tr>
<td>vfunc[5]</td>
<td>type_info::__do_upcast(__cxxabiv1::__c_class_type_info const*, void**) const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the __cxxabiv1::__enum_type_info class is described by Table 16-11.

Table 16-11 typeinfo for __cxxabiv1::__enum_type_info

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for __cxxabiv1::__enum_type_info</td>
</tr>
</tbody>
</table>

16.1.5.2 Interfaces for Class __cxxabiv1::__enum_type_info

An LSB conforming implementation shall provide the generic methods for Class __cxxabiv1::__enum_type_info specified in Table 16-12, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-12 libstdcxx - Class __cxxabiv1::__enum_type_info Function Interfaces

<table>
<thead>
<tr>
<th>Function Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>__cxxabiv1::__enum_type_info::~_enum_type_info()</td>
<td>(CXXABI-1.86)</td>
</tr>
<tr>
<td>__cxxabiv1::__enum_type_info::~_enum_type_info()</td>
<td>(CXXABI-1.86)</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class __cxxabiv1::__enum_type_info specified in Table 16-13, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-13 libstdcxx - Class __cxxabiv1::__enum_type_info Data Interfaces

<table>
<thead>
<tr>
<th>Typeinfo for __cxxabiv1::__enum_type_info(CXXABI_1.3) [CXXABI-1.86]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typeinfo name for __cxxabiv1::__enum_type_info(CXXABI_1.3) [CXXABI-1.86]</td>
</tr>
<tr>
<td>Vtable for __cxxabiv1::__enum_type_info(CXXABI_1.3) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

### 16.1.6 Class __cxxabiv1::__array_type_info

#### 16.1.6.1 Class data for __cxxabiv1::__array_type_info

The virtual table for the __cxxabiv1::__array_type_info class is described by Table 16-14.

### Table 16-14 Primary vtable for __cxxabiv1::__array_type_info

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __cxxabiv1::__array_type_info</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>__cxxabiv1::__array_type_info::~array_type_info()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>__cxxabiv1::__array_type_info::~array_type_info()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>type_info__is_pointer_p() const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>type_info__is_function_p() const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>type_info__do_catch(type_info const*, void**, unsigned int) const</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>type_info__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the __cxxabiv1::__array_type_info class is described by Table 16-15.

### Table 16-15 typeinfo for __cxxabiv1::__array_type_info

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for __cxxabiv1::__array_type_info</td>
</tr>
</tbody>
</table>

#### 16.1.6.2 Interfaces for Class __cxxabiv1::__array_type_info

An LSB conforming implementation shall provide the generic methods for Class __cxxabiv1::__array_type_info specified in Table 16-16, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-16 libstdcxx - Class __cxxabiv1::__array_type_info Function Interfaces

<table>
<thead>
<tr>
<th>__cxxabiv1::__array_type_info::~array_type_info()(CXXABI_1.3) [CXXABI-1.86]</th>
</tr>
</thead>
<tbody>
<tr>
<td>__cxxabiv1::__array_type_info::~array_type_info()(CXXABI_1.3) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class __cxxabiv1::__array_type_info specified in Table 16-17, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-17 | libstdcxx - Class __cxxabiv1::__array_type_info Data Interfaces |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __cxxabiv1::__array_type_info</td>
</tr>
<tr>
<td>vfunc[0]</td>
<td>__cxxabiv1::__array_type_info::~__array_type_info()</td>
</tr>
<tr>
<td>vfunc[1]</td>
<td>__cxxabiv1::__array_type_info::~__array_type_info()</td>
</tr>
<tr>
<td>vfunc[2]</td>
<td>type_info::__is_pointer_p() const</td>
</tr>
<tr>
<td>vfunc[3]</td>
<td>type_info::__is_function_p() const</td>
</tr>
<tr>
<td>vfunc[4]</td>
<td>__cxxabiv1::__array_type_info::do_cast(type_info const*, void**, unsigned int) const</td>
</tr>
<tr>
<td>vfunc[5]</td>
<td>__cxxabiv1::__class_type_info::do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&amp;) const</td>
</tr>
<tr>
<td>vfunc[6]</td>
<td>__cxxabiv1::__class_type_info::do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&amp;) const</td>
</tr>
</tbody>
</table>

The virtual table for the __cxxabiv1::__class_type_info class is described by Table 16-18.

### Table 16-18 | Primary vtable for __cxxabiv1::__class_type_info |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __cxxabiv1::__class_type_info</td>
</tr>
<tr>
<td>vfunc[0]</td>
<td>__cxxabiv1::__class_type_info::~__class_type_info()</td>
</tr>
<tr>
<td>vfunc[1]</td>
<td>__cxxabiv1::__class_type_info::~__class_type_info()</td>
</tr>
<tr>
<td>vfunc[2]</td>
<td>type_info::__is_pointer_p() const</td>
</tr>
<tr>
<td>vfunc[3]</td>
<td>type_info::__is_function_p() const</td>
</tr>
<tr>
<td>vfunc[4]</td>
<td>__cxxabiv1::__class_type_info::do_cast(type_info const*, void**, unsigned int) const</td>
</tr>
<tr>
<td>vfunc[5]</td>
<td>__cxxabiv1::__class_type_info::do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&amp;) const</td>
</tr>
<tr>
<td>vfunc[6]</td>
<td>__cxxabiv1::__class_type_info::do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&amp;) const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the __cxxabiv1::__class_type_info class is described by Table 16-19.

### Table 16-19 | typeinfo for __cxxabiv1::__class_type_info |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Vtable</td>
<td>vtable for __cxxabiv1::__class_type_info</td>
</tr>
<tr>
<td>Name</td>
<td>typeinfo name for __cxxabiv1::__class_type_info</td>
</tr>
</tbody>
</table>
16.1.7.2 Interfaces for Class __cxxabiv1::__class_type_info

An LSB conforming implementation shall provide the generic methods for Class __cxxabiv1::__class_type_info specified in Table 16-20, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-20 libstdcxx - Class __cxxabiv1::__class_type_info Function Interfaces

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>__cxxabiv1::__class_type_info::~__class_type_info()</td>
<td>Destructor</td>
</tr>
<tr>
<td>__cxxabiv1::__class_type_info::do_catch(type_info const*, void**, unsigned int) const</td>
<td>Catch function</td>
</tr>
<tr>
<td>__cxxabiv1::__class_type_info::do_upcast(type_info const*, void**, __cxxabiv1::__class_type_info::__upcast_result&amp;) const</td>
<td>Upcast function</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class __cxxabiv1::__class_type_info specified in Table 16-21, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-21 libstdcxx - Class __cxxabiv1::__class_type_info Data Interfaces

<table>
<thead>
<tr>
<th>Data Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for __cxxabiv1::__class_type_info</td>
<td>Type information</td>
</tr>
<tr>
<td>vtable for __cxxabiv1::__class_type_info</td>
<td>Virtual table</td>
</tr>
</tbody>
</table>

16.1.8 Class __cxxabiv1::__pbase_type_info

16.1.8.1 Class data for __cxxabiv1::__pbase_type_info

The virtual table for the __cxxabiv1::__pbase_type_info class is described by Table 16-22.

Table 16-22 Primary vtable for __cxxabiv1::__pbase_type_info

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Base Offset</td>
</tr>
<tr>
<td>0</td>
<td>Virtual Base Offset</td>
</tr>
<tr>
<td>0</td>
<td>RTTI</td>
</tr>
<tr>
<td>0</td>
<td>Vfunc[0]</td>
</tr>
<tr>
<td>0</td>
<td>Vfunc[1]</td>
</tr>
<tr>
<td>0</td>
<td>Vfunc[2]</td>
</tr>
<tr>
<td>0</td>
<td>Vfunc[3]</td>
</tr>
<tr>
<td>0</td>
<td>Vfunc[4]</td>
</tr>
<tr>
<td>0</td>
<td>Vfunc[5]</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
The Run Time Type Information for the __cxxabiv1::__pbase_type_info class is described by Table 16-23.

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for __cxxabiv1::__pbase_type_info</td>
</tr>
</tbody>
</table>

16.1.8.2 Interfaces for Class __cxxabiv1::__pbase_type_info

An LSB conforming implementation shall provide the generic methods for Class __cxxabiv1::__pbase_type_info specified in Table 16-24, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-24 libstdcxx - Class __cxxabiv1::__pbase_type_info Function Interfaces

| __cxxabiv1::__pbase_type_info::~__pbase_type_info()     | (CXXABI_1.3) [CXXABI-1.86] |
| __cxxabiv1::__pbase_type_info::~__pbase_type_info()     | (CXXABI_1.3) [CXXABI-1.86] |
| __cxxabiv1::__pbase_type_info::~__pbase_type_info()     | (CXXABI_1.3) [CXXABI-1.86] |
| __cxxabiv1::__pbase_type_info::~__pbase_type_info()     | (CXXABI_1.3) [CXXABI-1.86] |
| __cxxabiv1::__pbase_type_info::do_catch(type_info const*, void**, unsigned int) const | (CXXABI-1.86) |
| __cxxabiv1::__pbase_type_info::pointer_catch(__cxxabiv1::__pbase_type_info const*, void**, unsigned int) const | (CXXABI-1.86) |

An LSB conforming implementation shall provide the generic data interfaces for Class __cxxabiv1::__pbase_type_info specified in Table 16-25, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-25 libstdcxx - Class __cxxabiv1::__pbase_type_info Data Interfaces

| typeinfo for __cxxabiv1::__pbase_type_info(CXXABI_1.3) | [CXXABI-1.86] |
| typeinfo name for __cxxabiv1::__pbase_type_info(CXXABI_1.3) | [CXXABI-1.86] |
| vtable for __cxxabiv1::__pbase_type_info(CXXABI_1.3) | [CXXABI-1.86] |

16.1.9 Class __cxxabiv1::__pointer_type_info

16.1.9.1 Class data for __cxxabiv1::__pointer_type_info

The virtual table for the __cxxabiv1::__pointer_type_info class is described by Table 16-26.

Table 16-26 Primary vtable for __cxxabiv1::__pointer_type_info

| Base Offset | 0 |
| Virtual Base Offset | 0 |
| RTTI         | typeinfo for __cxxabiv1::__pointer_type_info |
| vfunc[0]:    | __cxxabiv1::__pointer_type_info::~__pointer_type_info() |
The Run Time Type Information for the `__cxxabiv1::__pointer_type_info` class is described by Table 16-27.

<table>
<thead>
<tr>
<th>Vfunc</th>
<th>Function Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>[vfunc[1]]</td>
<td><code>__cxxabiv1::__pointer_type_info::~__pointer_type_info()</code></td>
</tr>
<tr>
<td>[vfunc[2]]</td>
<td><code>__cxxabiv1::__pointer_type_info::__is_pointer_p() const</code></td>
</tr>
<tr>
<td>[vfunc[3]]</td>
<td><code>type_info::__is_function_p() const</code></td>
</tr>
<tr>
<td>[vfunc[4]]</td>
<td><code>__cxxabiv1::__pointer_type_info::__pointer_catch(__cxxabiv1::__pbase_type_info const*, void**, unsigned int) const</code></td>
</tr>
<tr>
<td>[vfunc[5]]</td>
<td><code>type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</code></td>
</tr>
<tr>
<td>[vfunc[6]]</td>
<td><code>__cxxabiv1::__pointer_type_info::__pointer_catch(__cxxabiv1::__pbase_type_info const*, void**, unsigned int) const</code></td>
</tr>
</tbody>
</table>

Table 16-27 typeinfo for `__cxxabiv1::__pointer_type_info`

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for <code>__cxxabiv1::__si_class_type_info</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for <code>__cxxabiv1::__pointer_type_info</code></td>
</tr>
</tbody>
</table>

16.1.9.2 Interfaces for Class `__cxxabiv1::__pointer_type_info`

An LSB conforming implementation shall provide the generic methods for Class `__cxxabiv1::__pointer_type_info` specified in Table 16-28, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-28 libstdcxx - Class `__cxxabiv1::__pointer_type_info` Function Interfaces

<table>
<thead>
<tr>
<th>Function Call</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>__cxxabiv1::__pointer_type_info::~__pointer_type_info()</code></td>
</tr>
<tr>
<td><code>__cxxabiv1::__pointer_type_info::__is_pointer_p() const</code></td>
</tr>
<tr>
<td><code>__cxxabiv1::__pointer_type_info::__pointer_catch(__cxxabiv1::__pbase_type_info const*, void**, unsigned int) const</code></td>
</tr>
<tr>
<td><code>type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</code></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class `__cxxabiv1::__pointer_type_info` specified in Table 16-29, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-29 libstdcxx - Class `__cxxabiv1::__pointer_type_info` Data Interfaces

<table>
<thead>
<tr>
<th>Data Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>typeinfo name for __cxxabiv1::__pointer_type_info(CXXABI_1.3)</code></td>
</tr>
<tr>
<td><code>vtable for __cxxabiv1::__pointer_type_info(CXXABI_1.3)</code></td>
</tr>
</tbody>
</table>
16.1.10 Class \_\_cxxabiv1::\_\_function\_type\_info

16.1.10.1 Class data for \_\_cxxabiv1::\_\_function\_type\_info

The virtual table for the \_\_cxxabiv1::\_\_function\_type\_info class is described by Table 16-30.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __cxxabiv1::__function_type_info</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>__cxxabiv1::__function_type_info::~__function_type_info()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>__cxxabiv1::__function_type_info::~__function_type_info()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>type_info::__is_pointer_p() const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>__cxxabiv1::__function_type_info::__is_function_p() const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>type_info::__do_catch(type_info const*, void**, unsigned_int) const</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the \_\_cxxabiv1::\_\_function\_type\_info class is described by Table 16-31.

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for __cxxabiv1::__function_type_info</td>
</tr>
</tbody>
</table>

16.1.10.2 Interfaces for Class \_\_cxxabiv1::\_\_function\_type\_info

An LSB conforming implementation shall provide the generic methods for Class \_\_cxxabiv1::\_\_function\_type\_info specified in Table 16-32, with the full mandatory functionality as described in the referenced underlying specification.

| \_\_cxxabiv1::\_\_function\_type\_info::\_\_function\_type\_info()(CXXABI\_L.86) | [CXXABI\_L.86] |
| \_\_cxxabiv1::\_\_function\_type\_info::\_\_function\_type\_info()(CXXABI\_L.86) | [CXXABI\_L.86] |
| \_\_cxxabiv1::\_\_function\_type\_info::\_\_function\_type\_info()(CXXABI\_L.86) | [CXXABI\_L.86] |
| \_\_cxxabiv1::\_\_function\_type\_info::\_\_is\_function\_p() const(CXXABI\_L.86) | [CXXABI\_L.86] |

An LSB conforming implementation shall provide the generic data interfaces for Class \_\_cxxabiv1::\_\_function\_type\_info specified in Table 16-33, with the full mandatory functionality as described in the referenced underlying specification.
### 16.1.11 Class __cxxabiv1::__si_class_type_info

#### 16.1.11.1 Class data for __cxxabiv1::__si_class_type_info

The virtual table for the __cxxabiv1::__si_class_type_info class is described by Table 16-34.

#### Table 16-34 Primary vtable for __cxxabiv1::__si_class_type_info

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __cxxabiv1::__si_class_type_info</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>__cxxabiv1::__si_class_type_info::__si_class_type_info()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>__cxxabiv1::__si_class_type_info::__si_class_type_info()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>type_info::__is_pointer_p() const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>type_info::__is_function_p() const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>__cxxabiv1::__class_type_info::__do_cast(type_info const*, void**, unsigned int) const</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**, __cxxabiv1::__class_type_info::upcast_result&amp;) const</td>
</tr>
<tr>
<td>vfunc[6]:</td>
<td>__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::upcast_result&amp;) const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the __cxxabiv1::__si_class_type_info class is described by Table 16-35.

#### Table 16-35 typeinfo for __cxxabiv1::__si_class_type_info

<table>
<thead>
<tr>
<th>Name</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>typeinfo name for __cxxabiv1::__si_class_type_info</td>
</tr>
</tbody>
</table>

#### 16.1.11.2 Interfaces for Class __cxxabiv1::__si_class_type_info

An LSB conforming implementation shall provide the generic methods for Class __cxxabiv1::__si_class_type_info specified in Table 16-36, with the full mandatory functionality as described in the referenced underlying specification.

#### Table 16-36 libstdcxx - Class __cxxabiv1::__si_class_type_info Function Interfaces

__cxxabiv1::__si_class_type_info::~__si_class_type_info__func((CXXABI_1.3) [CXXABI-1.86]

© 2015 Linux Foundation
An LSB conforming implementation shall provide the generic data interfaces for Class __cxxabiv1::__si_class_type_info specified in Table 16-37, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-37 libstdcxx - Class __cxxabiv1::__si_class_type_info Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>__cxxabiv1::__si_class_type_info::~__si_class_type_info()</td>
<td>[CXXABI-1.3]</td>
</tr>
<tr>
<td>__cxxabiv1::__si_class_type_info::~__si_class_type_info()</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>__cxxabiv1::__si_class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&amp;) const</td>
<td>[CXXABI-1.3] [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.12 Class __cxxabiv1::__vmi_class_type_info

16.1.12.1 Class data for __cxxabiv1::__vmi_class_type_info

The virtual table for the __cxxabiv1::__vmi_class_type_info class is described by Table 16-38.

Table 16-38 Primary vtable for __cxxabiv1::__vmi_class_type_info

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

| vfunc[0]:   | __cxxabiv1::__vmi_class_type_info::~__vmi_class_type_info() |
| vfunc[1]:   | __cxxabiv1::__vmi_class_type_info::~__vmi_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_cast(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::__vmi_class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&) const |

The Run Time Type Information for the __cxxabiv1::__vmi_class_type_info class is described by Table 16-39.

Table 16-39 typeinfo for __cxxabiv1::__vmi_class_type_info

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
</table>
16.1.12.2 Interfaces for Class __cxxabiv1::__vmi_class_type_info

An LSB conforming implementation shall provide the generic methods for Class __cxxabiv1::__vmi_class_type_info specified in Table 16-40, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-40 libstdcxx - Class __cxxabiv1::__vmi_class_type_info Function Interfaces

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>__cxxabiv1::__vmi_class_type_info:~__vmi_class_type_info</td>
<td>0</td>
<td>0</td>
<td>__cxxabiv1</td>
<td>__cxxabiv1::__vmi_class_type_info:~__vmi_class_type_info</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__cxxabiv1::__vmi_class_type_info:~__vmi_class_type_info</td>
<td>0</td>
<td>0</td>
<td>__cxxabiv1</td>
<td>__cxxabiv1::__vmi_class_type_info:~__vmi_class_type_info</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__cxxabiv1::__vmi_class_type_info:~__vmi_class_type_info</td>
<td>0</td>
<td>0</td>
<td>__cxxabiv1</td>
<td>__cxxabiv1::__vmi_class_type_info:~__vmi_class_type_info</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__cxxabiv1::__vmi_class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&amp;) const</td>
<td>0</td>
<td>0</td>
<td>__cxxabiv1</td>
<td>__cxxabiv1::__vmi_class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&amp;) const</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class __cxxabiv1::__vmi_class_type_info specified in Table 16-41, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-41 libstdcxx - Class __cxxabiv1::__vmi_class_type_info Data Interfaces

<table>
<thead>
<tr>
<th>Type Info</th>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]: __cxxabiv1::__vmi_class_type_info: __cxxabiv1::__vmi_class_type_info()</th>
<th>vfunc[1]: __cxxabiv1::__vmi_class_type_info: __cxxabiv1::__vmi_class_type_info()</th>
<th>vfunc[2]: type_info::__is_pointer_p() const</th>
<th>vfunc[3]: type_info::__is_function_p() const</th>
<th>vfunc[4]: type_info::__do_catch(type_info const*, void**, unsigned int) const</th>
<th>vfunc[5]: type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</th>
</tr>
</thead>
<tbody>
<tr>
<td>__cxxabiv1::__vmi_class_type_info</td>
<td>0</td>
<td>0</td>
<td>__cxxabiv1</td>
<td>__cxxabiv1::__vmi_class_type_info</td>
<td>__cxxabiv1::__vmi_class_type_info</td>
<td>type_info::__is_pointer_p() const</td>
<td>type_info::__is_function_p() const</td>
<td>type_info::__do_catch(type_info const*, void**, unsigned int) const</td>
<td>type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</td>
</tr>
</tbody>
</table>

16.1.13 Class __cxxabiv1::__fundamental_type_info

16.1.13.1 Class data for __cxxabiv1::__fundamental_type_info

The virtual table for the __cxxabiv1::__fundamental_type_info class is described by Table 16-42.

Table 16-42 Primary vtable for __cxxabiv1::__fundamental_type_info

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]: __cxxabiv1::__fundamental_type_info: __cxxabiv1::__fundamental_type_info()</th>
<th>vfunc[1]: __cxxabiv1::__fundamental_type_info: __cxxabiv1::__fundamental_type_info()</th>
<th>vfunc[2]: type_info::__is_pointer_p() const</th>
<th>vfunc[3]: type_info::__is_function_p() const</th>
<th>vfunc[4]: type_info::__do_catch(type_info const*, void**, unsigned int) const</th>
<th>vfunc[5]: type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>__cxxabiv1</td>
<td>__cxxabiv1::__fundamental_type_info</td>
<td>__cxxabiv1::__fundamental_type_info</td>
<td>type_info::__is_pointer_p() const</td>
<td>type_info::__is_function_p() const</td>
<td>type_info::__do_catch(type_info const*, void**, unsigned int) const</td>
<td>type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the __cxxabiv1::__fundamental_type_info class is described by Table 16-43

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for __cxxabiv1::__fundamental_type_info</td>
</tr>
</tbody>
</table>

### 16.1.13.2 Interfaces for Class __cxxabiv1::__fundamental_type_info

An LSB conforming implementation shall provide the generic methods for Class __cxxabiv1::__fundamental_type_info specified in Table 16-44, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function Interfaces</th>
<th>Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>__cxxabiv1::__fundamental_type_info::~__fundamental_type_info()(CXXABI_1.3)</td>
<td></td>
</tr>
<tr>
<td>__cxxabiv1::__fundamental_type_info::~__fundamental_type_info()(CXXABI_1.3)</td>
<td></td>
</tr>
<tr>
<td>__cxxabiv1::__fundamental_type_info::~__fundamental_type_info()(CXXABI_1.3)</td>
<td></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class __cxxabiv1::__fundamental_type_info specified in Table 16-45, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for __cxxabiv1::__fundamental_type_info(CXXABI_1.3) [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for __cxxabiv1::__fundamental_type_info(CXXABI_1.3) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for __cxxabiv1::__fundamental_type_info(CXXABI_1.3) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

### 16.1.14 Class __cxxabiv1::__pointer_to_member_type_info

#### 16.1.14.1 Class data for __cxxabiv1::__pointer_to_member_type_info

The virtual table for the __cxxabiv1::__pointer_to_member_type_info class is described by Table 16-46.

<table>
<thead>
<tr>
<th>Primary vtable for __cxxabiv1::__pointer_to_member_type_info</th>
<th>Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __cxxabiv1::__pointer_to_member_type_info</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>__cxxabiv1::__pointer_to_member_type_info::~__pointer_to_member_type_info()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>__cxxabiv1::__pointer_to_member_type_info::~__pointer_to_member_type_info()</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the __cxxabiv1::__pointer_to_member_type_info class is described by Table 16-47.

Table 16-47 typeinfo for __cxxabiv1::__pointer_to_member_type_info

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for __cxxabiv1::__pointer_to_member_type_info</td>
</tr>
</tbody>
</table>

### 16.1.14.2 Interfaces for Class __cxxabiv1::__pointer_to_member_type_info

An LSB conforming implementation shall provide the generic methods for Class __cxxabiv1::__pointer_to_member_type_info specified in Table 16-48, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>__cxxabiv1::__pointer_to_member_type_info::~__pointer_to_member_type_info()</td>
</tr>
<tr>
<td>__cxxabiv1::__pointer_to_member_type_info::~__pointer_to_member_type_info()</td>
</tr>
<tr>
<td>__cxxabiv1::__pointer_to_member_type_info::~__pointer_to_member_type_info()</td>
</tr>
<tr>
<td>__cxxabiv1::__pointer_to_member_type_info::__pointer_catch(__cxxabiv1::__pbasetype_info const*, void**, unsigned int) const</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class __cxxabiv1::__pointer_to_member_type_info specified in Table 16-49, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-49 libstdcxx - Class __cxxabiv1::__pointer_to_member_type_info Data Interfaces

<table>
<thead>
<tr>
<th>Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for __cxxabiv1::__pointer_to_member_type_info(CXXABI_1.3)</td>
</tr>
<tr>
<td>typeinfo name for __cxxabiv1::__pointer_to_member_type_info(CXXABI_1.3)</td>
</tr>
<tr>
<td>vtable for __cxxabiv1::__pointer_to_member_type_info(CXXABI_1.3)</td>
</tr>
</tbody>
</table>
16.1.15 Class __gnu_cxx::stdio_filebuf<char, char_traits<char> >

16.1.15.1 Interfaces for Class __gnu_cxx::stdio_filebuf<char, char_traits<char> >

No external methods are defined for libstdcxx - Class __gnu_cxx::stdio_filebuf<char, std::char_traits<char> > in this part of the specification. See also the relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for Class __gnu_cxx::stdio_filebuf<char, std::char_traits<char> > specified in Table 16-50, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for __gnu_cxx::stdio_filebuf&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for __gnu_cxx::stdio_filebuf&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.16 Class __gnu_cxx::stdio_filebuf<wchar_t, char_traits<wchar_t> >

16.1.16.1 Interfaces for Class __gnu_cxx::stdio_filebuf<wchar_t, char_traits<wchar_t> >

No external methods are defined for libstdcxx - Class __gnu_cxx::stdio_filebuf<wchar_t, std::char_traits<wchar_t> > in this part of the specification. See also the relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for Class __gnu_cxx::stdio_filebuf<wchar_t, std::char_traits<wchar_t> > specified in Table 16-51, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for __gnu_cxx::stdio_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for __gnu_cxx::stdio_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.17 Class __gnu_cxx::__pool_alloc_base

16.1.17.1 Interfaces for Class __gnu_cxx::__pool_alloc_base

An LSB conforming implementation shall provide the generic methods for Class __gnu_cxx::__pool_alloc_base specified in Table 16-52, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>__gnu_cxx::__pool_alloc_base::_M_get_mutex()(GLIBCXX_3.4.2) [LSB]</td>
</tr>
</tbody>
</table>
16.1.18 Class __gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >

16.1.18.1 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 16-53.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt; &gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt; &gt;::~stdio_sync_filebuf()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt; &gt;::~stdio_sync_filebuf()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::imbue(locale const&amp;)</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt; &gt;::seekpos(fpos&lt;_mbstate_t&gt;, _Ios_Openmode)</td>
</tr>
<tr>
<td>vfunc[6]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt; &gt;::sync()</td>
</tr>
<tr>
<td>vfunc[7]:</td>
<td>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::showmanyc()</td>
</tr>
<tr>
<td>vfunc[8]:</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>vfunc[9]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt; &gt;::underflow()</td>
</tr>
<tr>
<td>vfunc[10]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt; &gt;::uflow()</td>
</tr>
<tr>
<td>vfunc[11]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt; &gt;::pbackfail(int)</td>
</tr>
<tr>
<td>vfunc[12]:</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>vfunc[13]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt; &gt;::overflow(int)</td>
</tr>
</tbody>
</table>

16.1.18.2 Interfaces for Class __gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >

An LSB conforming implementation shall provide the generic methods for Class __gnu_cxx::stdio_sync_filebuf<char, std::char_traits<char> > specified in Table 16-54, with the full mandatory functionality as described in the referenced underlying specification.
An LSB conforming implementation shall provide the generic data interfaces for Class
__gnu_cxx::stdio_sync_filebuf<char, std::char_traits<char> > specified in Table 16-55,
with the full mandatory functionality as described in the referenced underlying specifi-
cation.

16.1.19 Class
__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >

16.1.19.1 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 16-56

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __gnu_cxx::stdio_sync_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~stdio_sync_filebuf()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~stdio_sync_filebuf()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::imbue(locale const&amp;)</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::seekpos(fpos&lt;__mbstate_t&gt;, _Ios_Openmode)</td>
</tr>
<tr>
<td>vfunc[6]:</td>
<td>__gnu_cxx::stdio_sync_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::sync()</td>
</tr>
</tbody>
</table>
16.1.19.2 Interfaces for Class
__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the generic methods for Class
__gnu_cxx::stdio_sync_filebuf<wchar_t, std::char_traits<wchar_t> > specified in Table
16-57, with the full mandatory functionality as described in the referenced underlying
specification.

Table 16-57 libstdcxx - Class __gnu_cxx::stdio_sync_filebuf<wchar_t,
char_traits<wchar_t> > Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
</table>
| basic_streambuf<wchar_t,
  char_traits<wchar_t> >::showmanyc() | vfunc[7]:                                        |
| __gnu_cxx::stdio_sync_filebuf<wchar_t,
  char_traits<wchar_t> >::underflow() | vfunc[9]:                                        |
| __gnu_cxx::stdio_sync_filebuf<wchar_t,
  char_traits<wchar_t> >::uflow() | vfunc[10]:                                       |
| __gnu_cxx::stdio_sync_filebuf<wchar_t,
  char_traits<wchar_t> >::pbackfail(unsigned int) | vfunc[11]:                                       |
| See architecture specific part. | vfunc[12]:                                       |
| __gnu_cxx::stdio_sync_filebuf<wchar_t,
  char_traits<wchar_t> >::overflow(unsigned int) | vfunc[13]:                                       |

An LSB conforming implementation shall provide the generic data interfaces for Class
__gnu_cxx::stdio_sync_filebuf<wchar_t, std::char_traits<wchar_t> > specified in Table
16-58, with the full mandatory functionality as described in the referenced underlying
specification.

Table 16-58 libstdcxx - Class __gnu_cxx::stdio_sync_filebuf<wchar_t,
char_traits<wchar_t> > Data Interfaces

<table>
<thead>
<tr>
<th>Data Interface</th>
<th>Description</th>
</tr>
</thead>
</table>
| typeinfo for __gnu_cxx::stdio_sync_filebuf<wchar_t,
  char_traits<wchar_t> > | vfunc[0]:                                        |
| typeinfo name for __gnu_cxx::stdio_sync_filebuf<wchar_t,
  char_traits<wchar_t> > | vfunc[1]:                                        |
| vtable for __gnu_cxx::stdio_sync_filebuf<wchar_t,
  char_traits<wchar_t> > | vfunc[2]:                                        |

16.1.20 Class exception

16.1.20.1 Class data for exception

The virtual table for the std::exception class is described by Table 16-59

Table 16-59 Primary vtable for exception

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for exception</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>exception::~exception()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>exception::~exception()</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the std::exception class is described by Table 16-60.

### Table 16-60 typeinfo for exception

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for exception</td>
</tr>
</tbody>
</table>

#### 16.1.20.2 Interfaces for Class exception

An LSB conforming implementation shall provide the generic methods for Class std::exception specified in Table 16-61, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-61 libstdcxx - Class exception Function Interfaces

| exception::what() const(GLIBCXX_3.4) | [ISOCXX] |
| exception::~exception()(GLIBCXX_3.4) | [ISOCXX] |
| exception::~exception()(GLIBCXX_3.4) | [ISOCXX] |
| exception::~exception()(GLIBCXX_3.4) | [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::exception specified in Table 16-62, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-62 libstdcxx - Class exception Data Interfaces

| typeinfo for exception(GLIBCXX_3.4) | [CXXABI-1.86] |
| vtable for exception(GLIBCXX_3.4)   | [CXXABI-1.86] |

#### 16.1.21 Class bad_typeid

##### 16.1.21.1 Class data for bad_typeid

The virtual table for the std::bad_typeid class is described by Table 16-63.

### Table 16-63 Primary vtable for bad_typeid

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for bad_typeid</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>bad_typeid::~bad_typeid()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>bad_typeid::~bad_typeid()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>exception::what() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::bad_typeid class is described by Table 16-64.

### Table 16-64 typeinfo for bad_typeid

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for bad_typeid</td>
</tr>
</tbody>
</table>
16 Libraries

LSB Core - Generic 5.0

16.1.21.2 Interfaces for Class bad_typeid

An LSB conforming implementation shall provide the generic methods for Class std::bad_typeid specified in Table 16-65, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-65 libstdcxx - Class bad_typeid Function Interfaces

| bad_typeid::~bad_typeid() (GLIBCXX_3.4) [ISOCXX] |
| bad_typeid::~bad_typeid() (GLIBCXX_3.4) [ISOCXX] |
| bad_typeid::~bad_typeid() (GLIBCXX_3.4) [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::bad_typeid specified in Table 16-66, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-66 libstdcxx - Class bad_typeid Data Interfaces

| typeinfo for bad_typeid(GLIBCXX_3.4) [CXXABI-1.86] |
| typeinfo name for bad_typeid(GLIBCXX_3.4) [CXXABI-1.86] |
| vtable for bad_typeid(GLIBCXX_3.4) [CXXABI-1.86] |

16.1.22 Class logic_error

16.1.22.1 Class data for logic_error

The virtual table for the std::logic_error class is described by Table 16-67.

Table 16-67 Primary vtable for logic_error

| Base Offset | 0 |
| Virtual Base Offset | 0 |
| RTTI |
| vfunc[0]: logic_error::~logic_error() |
| vfunc[1]: logic_error::~logic_error() |
| vfunc[2]: logic_error::what() const |

The Run Time Type Information for the std::logic_error class is described by Table 16-68.

Table 16-68 typeinfo for logic_error

| Base Vtable |
| vtable for __cxaabi::__si_class_type_info |
| Name |
| typename name for logic_error |

16.1.22.2 Interfaces for Class logic_error

An LSB conforming implementation shall provide the generic methods for Class std::logic_error specified in Table 16-69, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-69 libstdcxx - Class logic_error Function Interfaces

| logic_error::what() const(GLIBCXX_3.4) [ISOCXX] |
| logic_error::logic_error(basic_string<char, char_traits<char>, allocator<char>> const&) (GLIBCXX_3.4) [ISOCXX] |
| logic_error::logic_error(basic_string<char, char_traits<char>, allocator<char>> const&) (GLIBCXX_3.4) [ISOCXX] |
An LSB conforming implementation shall provide the generic data interfaces for Class std::logic_error specified in Table 16-70, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-70** libstdcxx - Class logic_error Data Interfaces

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for logic_error(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for logic_error(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for logic_error(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

### 16.1.23 Class range_error

#### 16.1.23.1 Class data for range_error

The virtual table for the std::range_error class is described by Table 16-71

**Table 16-71** Primary vtable for range_error

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for range_error</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>range_error::~range_error()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>range_error::~range_error()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>runtime_error::what() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::range_error class is described by Table 16-72

**Table 16-72** typeinfo for range_error

<table>
<thead>
<tr>
<th></th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for range_error</td>
</tr>
</tbody>
</table>

#### 16.1.23.2 Interfaces for Class range_error

An LSB conforming implementation shall provide the generic methods for Class std::range_error specified in Table 16-73, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-73** libstdcxx - Class range_error Function Interfaces

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>range_error::range_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>range_error::range_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>range_error::~range_error()(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>range_error::~range_error()(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::range_error specified in Table 16-74, with the full mandatory functionality as described in the referenced underlying specification.
16 Libraries

<table>
<thead>
<tr>
<th>Table 16-74 libstdcxx - Class range_error Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for range_error(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for range_error(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for range_error(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.24 Class domain_error

16.1.24.1 Class data for domain_error

The virtual table for the std::domain_error class is described by Table 16-75.

<table>
<thead>
<tr>
<th>Table 16-75 Primary vtable for domain_error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Offset</td>
</tr>
<tr>
<td>Virtual Base Offset</td>
</tr>
<tr>
<td>RTTI</td>
</tr>
<tr>
<td>vfunc[0]:</td>
</tr>
<tr>
<td>vfunc[1]:</td>
</tr>
<tr>
<td>vfunc[2]:</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::domain_error class is described by Table 16-76.

<table>
<thead>
<tr>
<th>Table 16-76 typeinfo for domain_error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Vtable</td>
</tr>
<tr>
<td>Name</td>
</tr>
</tbody>
</table>

16.1.24.2 Interfaces for Class domain_error

An LSB conforming implementation shall provide the generic methods for Class std::domain_error specified in Table 16-77, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 16-77 libstdcxx - Class domain_error Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain_error::domain_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>domain_error::domain_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>domain_error::~domain_error()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>domain_error::~domain_error()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::domain_error specified in Table 16-78, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 16-78 libstdcxx - Class domain_error Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for domain_error(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for domain_error(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for domain_error(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>
16.1.25 Class length_error

16.1.25.1 Class data for length_error

The virtual table for the std::length_error class is described by Table 16-79.

Table 16-79 Primary vtable for length_error
<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for length_error</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>length_error::~length_error()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>length_error::~length_error()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>logic_error::what() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::length_error class is described by Table 16-80.

Table 16-80 typeinfo for length_error
<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for length_error</td>
</tr>
</tbody>
</table>

16.1.25.2 Interfaces for Class length_error

An LSB conforming implementation shall provide the generic methods for Class std::length_error specified in Table 16-81, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-81 libstdcxx - Class length_error Function Interfaces
| length_error::length_error(basic_string<char, char_traits<char>, allocator<char> > const&)(GLIBCXX_3.4) [ISOCXX] |
| length_error::length_error(basic_string<char, char_traits<char>, allocator<char> > const&)(GLIBCXX_3.4) [ISOCXX] |
| length_error::~length_error()(GLIBCXX_3.4) [ISOCXX] |
| length_error::~length_error()(GLIBCXX_3.4) [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::length_error specified in Table 16-82, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-82 libstdcxx - Class length_error Data Interfaces
| typeinfo for length_error(GLIBCXX_3.4) [CXXABI-1.86] |
| typeinfo name for length_error(GLIBCXX_3.4) [CXXABI-1.86] |
| vtable for length_error(GLIBCXX_3.4) [CXXABI-1.86] |

16.1.26 Class out_of_range

16.1.26.1 Class data for out_of_range

The virtual table for the std::out_of_range class is described by Table 16-83.

Table 16-83 Primary vtable for out_of_range
| Base Offset | 0 |
| Virtual Base Offset | 0 |
The Run Time Type Information for the std::out_of_range class is described by Table 16-84.

Table 16-84 typeinfo for out_of_range

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for out_of_range</td>
</tr>
</tbody>
</table>

16.1.26.2 Interfaces for Class out_of_range

An LSB conforming implementation shall provide the generic methods for Class std::out_of_range specified in Table 16-85, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-85 libstdcxx - Class out_of_range Function Interfaces

| out_of_range::out_of_range(basic_string<char, char_traits<char>, allocator<char> > const&) [GLIBCXX_3.4] [ISOcxx] |
| out_of_range::out_of_range(basic_string<char, char_traits<char>, allocator<char> > const&) [GLIBCXX_3.4] [ISOcxx] |
| out_of_range::out_of_range()(GLIBCXX_3.4) [ISOcxx] |
| out_of_range::~out_of_range()(GLIBCXX_3.4) [ISOcxx] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::out_of_range specified in Table 16-86, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-86 libstdcxx - Class out_of_range Data Interfaces

| typeinfo for out_of_range(GLIBCXX_3.4) [CXXABI-1.86] |
| typeinfo name for out_of_range(GLIBCXX_3.4) [CXXABI-1.86] |
| vtable for out_of_range(GLIBCXX_3.4) [CXXABI-1.86] |

16.1.27 Class bad_exception

16.1.27.1 Class data for bad_exception

The virtual table for the std::bad_exception class is described by Table 16-87.

Table 16-87 Primary vtable for bad_exception

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for bad_exception</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>bad_exception::~bad_exception()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>bad_exception::~bad_exception()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>exception::what() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::bad_exception class is described by Table 16-88.
16.1.27.2 Interfaces for Class bad_exception

An LSB conforming implementation shall provide the generic methods for Class std::bad_exception specified in Table 16-89, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-89 libstdcxx - Class bad_exception Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bad_exception::~bad_exception() (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>bad_exception::~bad_exception() (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>bad_exception::~bad_exception() (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::bad_exception specified in Table 16-90, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-90 libstdcxx - Class bad_exception Data Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for bad_exception(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for bad_exception(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for bad_exception(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.28 Class runtime_error

16.1.28.1 Class data for runtime_error

The virtual table for the std::runtime_error class is described by Table 16-91.

Table 16-91 Primary vtable for runtime_error

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Base Offset</td>
</tr>
<tr>
<td>0</td>
<td>Virtual Base Offset</td>
</tr>
<tr>
<td>0</td>
<td>RTTI</td>
</tr>
<tr>
<td>runtime_error::~runtime_error()</td>
<td>vfunc[0]:</td>
</tr>
<tr>
<td>runtime_error::~runtime_error()</td>
<td>vfunc[1]:</td>
</tr>
<tr>
<td>runtime_error::what() const</td>
<td>vfunc[2]:</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::runtime_error class is described by Table 16-92.

Table 16-92 typeinfo for runtime_error

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vtable for _cxxabiv1::__si_class_type_info</td>
<td>Base Vtable</td>
</tr>
<tr>
<td>typeinfo name for runtime_error</td>
<td>Name</td>
</tr>
</tbody>
</table>

16.1.28.2 Interfaces for Class runtime_error

An LSB conforming implementation shall provide the generic methods for Class std::runtime_error specified in Table 16-93, with the full mandatory functionality as described in the referenced underlying specification.
16 Libraries

Table 16-93 libstdcxx - Class runtime_error Function Interfaces

<table>
<thead>
<tr>
<th>Function Description</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>runtime_error::what() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>runtime_error::runtime_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;) (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>runtime_error::runtime_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;) (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>runtime_error::~runtime_error() (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>runtime_error::~runtime_error() (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>runtime_error::~runtime_error() (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::runtime_error specified in Table 16-94, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-94 libstdcxx - Class runtime_error Data Interfaces

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for runtime_error(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for runtime_error(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for runtime_error(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.29 Class overflow_error

16.1.29.1 Class data for overflow_error

The virtual table for the std::overflow_error class is described by Table 16-95.

Table 16-95 Primary vtable for overflow_error

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for overflow_error</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>overflow_error::~overflow_error()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>overflow_error::~overflow_error()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>runtime_error::what() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::overflow_error class is described by Table 16-96.

Table 16-96 typeinfo for overflow_error

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for overflow_error</td>
</tr>
</tbody>
</table>

16.1.29.2 Interfaces for Class overflow_error

An LSB conforming implementation shall provide the generic methods for Class std::overflow_error specified in Table 16-97, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-97 libstdcxx - Class overflow_error Function Interfaces

<table>
<thead>
<tr>
<th>Function Description</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>overflow_error::overflow_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;) (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>overflow_error::overflow_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;) (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>overflow_error::overflow_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;) (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>overflow_error::~overflow_error() (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class std::overflow_error specified in Table 16-98, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-98 libstdcxx - Class overflow_error Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for overflow_error(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for overflow_error(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for overflow_error(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.30 Class underflow_error

16.1.30.1 Class data for underflow_error

The virtual table for the std::underflow_error class is described by Table 16-99.

Table 16-99 Primary vtable for underflow_error

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for underflow_error</td>
</tr>
<tr>
<td>vfunc[0]</td>
<td>underflow_error::~underflow_error()</td>
</tr>
<tr>
<td>vfunc[1]</td>
<td>underflow_error::underflow_error()</td>
</tr>
<tr>
<td>vfunc[2]</td>
<td>runtime_error::what() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::underflow_error class is described by Table 16-100.

Table 16-100 typeinfo for underflow_error

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Vtable</td>
<td>vtable for __cxxabiv1::__si_class_type_info</td>
</tr>
<tr>
<td></td>
<td>typeinfo name for underflow_error</td>
</tr>
</tbody>
</table>

16.1.30.2 Interfaces for Class underflow_error

An LSB conforming implementation shall provide the generic methods for Class std::underflow_error specified in Table 16-101, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-101 libstdcxx - Class underflow_error Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>underflow_error::underflow_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>underflow_error::underflow_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>underflow_error::~underflow_error()</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>underflow_error::~underflow_error()</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::underflow_error specified in Table 16-102, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-102 libstdcxx - Class underflow_error Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for underflow_error(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for underflow_error(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
16 Libraries

LSB Core - Generic 5.0

16.1.31 Class invalid_argument

16.1.31.1 Class data for invalid_argument

The virtual table for the std::invalid_argument class is described by Table 16-103.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for invalid_argument</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>invalid_argument::~invalid_argument()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>invalid_argument::~invalid_argument()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>logic_error::what() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::invalid_argument class is described by Table 16-104.

<table>
<thead>
<tr>
<th>Name</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for invalid_argument</td>
</tr>
</tbody>
</table>

16.1.31.2 Interfaces for Class invalid_argument

An LSB conforming implementation shall provide the generic methods for Class std::invalid_argument specified in Table 16-105, with the full mandatory functionality as described in the referenced underlying specification.

| invalid_argument::invalid_argument(basic_string<char, char_traits<char>, allocator<char>> const&)(GLIBCXX_3.4) | [ISOCXX] |
| invalid_argument::invalid_argument(basic_string<char, char_traits<char>, allocator<char>> const&)(GLIBCXX_3.4) | [ISOCXX] |
| invalid_argument::~invalid_argument()(GLIBCXX_3.4) | [ISOCXX] |
| invalid_argument::~invalid_argument()(GLIBCXX_3.4) | [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::invalid_argument specified in Table 16-106, with the full mandatory functionality as described in the referenced underlying specification.

| vtable for invalid_argument(GLIBCXX_3.4) | [CXXABI-1.86] |
| typeinfo name for invalid_argument(GLIBCXX_3.4) | [CXXABI-1.86] |
| vtable for invalid_argument(GLIBCXX_3.4) | [CXXABI-1.86] |

16.1.32 Class bad_cast

16.1.32.1 Class data for bad_cast

The virtual table for the std::bad_cast class is described by Table 16-107.
Table 16-107 Primary vtable for bad_cast

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for bad_cast</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>bad_cast::~bad_cast()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>bad_cast::~bad_cast()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>exception::what() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::bad_cast class is described by Table 16-108.

Table 16-108 typeinfo for bad_cast

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for bad_cast</td>
</tr>
</tbody>
</table>

16.1.32.2 Interfaces for Class bad_cast

An LSB conforming implementation shall provide the generic methods for Class std::bad_cast specified in Table 16-109, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-109 libstdcxx - Class bad_cast Function Interfaces

| bad_cast::~bad_cast() | [ISOCXX] |
| bad_cast::~bad_cast() | [ISOCXX] |
| bad_cast::~bad_cast() | [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::bad_cast specified in Table 16-110, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-110 libstdcxx - Class bad_cast Data Interfaces

| typeinfo for bad_cast(GLIBCXX_3.4) | [CXXABI-1.86] |
| typeinfo name for bad_cast(GLIBCXX_3.4) | [CXXABI-1.86] |
| vtable for bad_cast(GLIBCXX_3.4) | [CXXABI-1.86] |

16.1.33 Class bad_alloc

16.1.33.1 Class data for bad_alloc

The virtual table for the std::bad_alloc class is described by Table 16-111.

Table 16-111 Primary vtable for bad_alloc

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for bad_alloc</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>bad_alloc::~bad_alloc()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>bad_alloc::~bad_alloc()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>exception::what() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::bad_alloc class is described by Table 16-112.
16 Libraries

Table 16-112 typeinfo for bad_alloc

<table>
<thead>
<tr>
<th>Name</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
<th>typeinfo name for bad_alloc</th>
</tr>
</thead>
</table>

16.1.33 Interfaces for Class bad_alloc

An LSB conforming implementation shall provide the generic methods for Class std::bad_alloc specified in Table 16-113, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-113 libstdcxx - Class bad_alloc Function Interfaces

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>bad_alloc::~bad_alloc()</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>bad_alloc::~bad_alloc()</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>bad_alloc::~bad_alloc()</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
<td></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::bad_alloc specified in Table 16-114, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-114 libstdcxx - Class bad_alloc Data Interfaces

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for bad_alloc(GLIBCXX_3.4)</td>
<td>CXXABI-1.86</td>
<td></td>
</tr>
<tr>
<td>typeinfo name for bad_alloc(GLIBCXX_3.4)</td>
<td>CXXABI-1.86</td>
<td></td>
</tr>
<tr>
<td>vtable for bad_alloc(GLIBCXX_3.4)</td>
<td>CXXABI-1.86</td>
<td></td>
</tr>
</tbody>
</table>

16.1.34 struct __numeric_limits_base

16.1.34.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct __numeric_limits_base specified in Table 16-115, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-115 libstdcxx - struct __numeric_limits_base Data Interfaces

| __numeric_limits_base::has_denorm(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::is_bounded(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::is_integer(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::round_style(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::has_infinity(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::max_exponent(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::min_exponent(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::has_quiet_NaN(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::is_specialized(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::max_exponent10(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::min_exponent10(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::has_denorm_loss(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::tinyness_before(GLIBCXX_3.4) | ISOCXX |
| __numeric_limits_base::has_signaling_NaN(GLIBCXX_3.4) | ISOCXX |
16.1.35 struct numeric_limits<long double>

16.1.35.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<long double> specified in Table 16-116, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-116 libstdcxx - struct numeric_limits<long double> Data Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;long double&gt;::has_denorm(GLIBCXX_3.4)</td>
<td>has denorm</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::is_bounded(GLIBCXX_3.4)</td>
<td>is bounded</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::is_integer(GLIBCXX_3.4)</td>
<td>is integer</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::round_style(GLIBCXX_3.4)</td>
<td>round_style</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::has_infinity(GLIBCXX_3.4)</td>
<td>has infinity</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::max_exponent(GLIBCXX_3.4)</td>
<td>max exponent</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::min_exponent(GLIBCXX_3.4)</td>
<td>min exponent</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::is_specialized(GLIBCXX_3.4)</td>
<td>is_specialized</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::max_exponent10(GLIBCXX_3.4)</td>
<td>max exponent10</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::min_exponent10(GLIBCXX_3.4)</td>
<td>min exponent10</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::has_denorm_loss(GLIBCXX_3.4)</td>
<td>has denorm loss</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::tinyness_before(GLIBCXX_3.4)</td>
<td>tinyness_before</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::has_signaling_NaN(GLIBCXX_3.4)</td>
<td>has signaling NaN</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::radix(GLIBCXX_3.4)</td>
<td>radix</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::traps(GLIBCXX_3.4)</td>
<td>traps</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::digits(GLIBCXX_3.4)</td>
<td>digits</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::digits10(GLIBCXX_3.4)</td>
<td>digits10</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::is_exact(GLIBCXX_3.4)</td>
<td>is_exact</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::is_iec559(GLIBCXX_3.4)</td>
<td>is_iec559</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::is_modulo(GLIBCXX_3.4)</td>
<td>is modulo</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long double&gt;::is_signed(GLIBCXX_3.4)</td>
<td>is signed</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>
16.1.36 struct numeric_limits<long long>

16.1.36.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<long long> specified in Table 16-117, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-117 libstdcxx - struct numeric_limits<long long> Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>C++ Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;long long&gt;::has_denorm(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::is_bounded(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::is_integer(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::round_style(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::has_infinity(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::max_exponent(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::min_exponent(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::has_quiet_NaN(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::is_specialized(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::max_exp10(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::min_exp10(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::has_denorm_loss(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::tinyness_before(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::has_signaling_NaN(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::radix(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::traps(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::digits(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::digits10(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::is_exact(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::is_iec559(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::is_modulo(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;long long&gt;::is_signed(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.37 struct numeric_limits<unsigned long long>

16.1.37.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<unsigned long long> specified in Table 16-118, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-118 libstdcxx - struct numeric_limits<unsigned long long> Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>C++ Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;unsigned long long&gt;::has_denorm(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned long long&gt;::is_bounded(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>
16.1.38 struct numeric_limits<float>

16.1.38.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<float> specified in Table 16-119, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-119 libstdcxx - struct numeric_limits<float> Data Interfaces

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;float&gt;::has_denorm(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::is_bounded(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::is_integer(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::round_style(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::has_infinity(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::max_exponent(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::min_exponent(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::is_specialized(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::max_exponent10(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::min_exponent10(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::tinyness_before(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>numeric_limits&lt;float&gt;::is_signed(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
</tbody>
</table>
16.1.39 struct numeric_limits<double>

16.1.39.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<double> specified in Table 16-120, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-120 libstdcxx - struct numeric_limits<double> Data Interfaces

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>has_denorm(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_bounded(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_integer(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>round_style(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>has_infinity(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>max_exponent(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>min_exponent(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>has_quiet_NaN(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_specialized(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>max_exponent10(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>min_exponent10(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>has_denorm_loss(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>tinyness_before(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>has_signaling_NaN(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>radix(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>traps(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>digits(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>digits10(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_exact(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_iec559(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_modulo(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_signed(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>
16.1.40 struct numeric_limits<short>

16.1.40.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<short> specified in Table 16-121, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 16-121 libstdcxx - struct numeric_limits&lt;short&gt; Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;short&gt;::has_denorm(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::is_bounded(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::is_integer(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::round_style(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::has_infinity(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::max_exponent(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::min_exponent(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::is_specialized(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::max_exponent10(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::min_exponent10(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::tinyness_before(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::radix(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::traps(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::digits(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::digits10(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::is_exact(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::is_iec559(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::is_modulo(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;short&gt;::is_signed(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.41 struct numeric_limits<unsigned short>

16.1.41.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<unsigned short> specified in Table 16-122, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 16-122 libstdcxx - struct numeric_limits&lt;unsigned short&gt; Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;unsigned short&gt;::has_denorm(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned short&gt;::is_bounded(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned short&gt;::is_integer(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>
### 16.1.42 struct `numeric_limits<int>`

#### 16.1.42.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct `numeric_limits<int>` specified in Table 16-123, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>numeric_limits&lt;int&gt;::has_denorm</code></td>
<td>Has denorm</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::is_bounded</code></td>
<td>Is bounded</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::is_integer</code></td>
<td>Is integer</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::round_style</code></td>
<td>Round style</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::has_infinity</code></td>
<td>Has infinity</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::max_exponent</code></td>
<td>Max exponent</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::min_exponent</code></td>
<td>Min exponent</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::has_quiet_NaN</code></td>
<td>Has quiet NaN</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::is_specialized</code></td>
<td>Is specialized</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::max_exponent10</code></td>
<td>Max exponent10</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::min_exponent10</code></td>
<td>Min exponent10</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::has_denorm_loss</code></td>
<td>Has denorm loss</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::tinyness_before</code></td>
<td>Tinyness before</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::has_signaling_NaN</code></td>
<td>Has signaling NaN</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::radix</code></td>
<td>Radix</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::traps</code></td>
<td>Traps</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::digits</code></td>
<td>Digits</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::digits10</code></td>
<td>Digits10</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::is_exact</code></td>
<td>Is exact</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::is_iec559</code></td>
<td>Is IEC 559</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::is_modulo</code></td>
<td>Is modulo</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::is_signed</code></td>
<td>Is signed</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td><code>numeric_limits&lt;int&gt;::radix</code></td>
<td>Radix</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
</tbody>
</table>
16.1.43 struct numeric_limits<unsigned int>

16.1.43.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<unsigned int> specified in Table 16-124, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-124 libstdcxx - struct numeric_limits<unsigned int> Data Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::has_denorm</td>
<td>has denormalized value</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::is_bounded</td>
<td>is bounded</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::is_integer</td>
<td>is integer</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::round_style</td>
<td>round style</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::has_infinity</td>
<td>has infinity</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::max_exponent</td>
<td>maximum exponent</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::min_exponent</td>
<td>minimum exponent</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::has_quiet_NaN</td>
<td>has quiet NaN</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::is_specialized</td>
<td>is specialized</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::max_exponent10</td>
<td>maximum exponent10</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::min_exponent10</td>
<td>minimum exponent10</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::has_denorm_loss</td>
<td>has denormal loss</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::tinyness_before</td>
<td>is before tiny</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::has_signaling_NaN</td>
<td>has signaling NaN</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::radix</td>
<td>radix</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::traps</td>
<td>traps</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::digits</td>
<td>digits</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::digits10</td>
<td>digits10</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::is_exact</td>
<td>is exact</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::is_iec559</td>
<td>is IEC 559</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::is_modulo</td>
<td>is modulo</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned int&gt;::is_signed</td>
<td>is signed</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.44 struct numeric_limits<long>

16.1.44.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<long> specified in Table 16-125, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-125 libstdcxx - struct numeric_limits<long> Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;long&gt;::has_denorm</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::is_bounded</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::is_integer</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::round_style</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::hasInfinity</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::max_exponent</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::min_exponent</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::has_quiet_NaN</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::is_specialized</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::max_exponent10</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::min_exponent10</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::has_denorm_loss</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::tinyness_before</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::has_signaling_NaN</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::radix</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::traps</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::digits</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::digits10</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::is_exact</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::is_iec559</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::is_modulo</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;long&gt;::is_signed</td>
<td>GLIBCXX_3.4</td>
</tr>
</tbody>
</table>

16.1.45 struct numeric_limits<unsigned long>

16.1.45.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<unsigned long> specified in Table 16-126, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-126 libstdcxx - struct numeric_limits<unsigned long> Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;unsigned long&gt;::has_denorm</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned long&gt;::is_bounded</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned long&gt;::is_integer</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned long&gt;::round_style</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned long&gt;::hasInfinity</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned long&gt;::max_exponent</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned long&gt;::min_exponent</td>
<td>GLIBCXX_3.4</td>
</tr>
</tbody>
</table>
16.1.46 struct numeric_limits<wchar_t>

16.1.46.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<wchar_t> specified in Table 16-127, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>has_denorm</td>
<td>Checks if the floating point type has denorms.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_bounded</td>
<td>Checks if the floating point type is bounded.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_integer</td>
<td>Checks if the floating point type is an integer.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>round_style</td>
<td>Checks the rounding style of the floating point type.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>has_infinity</td>
<td>Checks if the floating point type has infinities.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>max_exponent</td>
<td>Checks the maximum exponent of the floating point type.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>min_exponent</td>
<td>Checks the minimum exponent of the floating point type.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>has_quiet_NaN</td>
<td>Checks if the floating point type has quiet NaNs.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_specialized</td>
<td>Checks if the floating point type is specialized.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>max_exponent10</td>
<td>Checks the maximum exponent in base 10 of the floating point type.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>min_exponent10</td>
<td>Checks the minimum exponent in base 10 of the floating point type.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>has_denorm_loss</td>
<td>Checks if the floating point type has denorm loss.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>tinyness_before</td>
<td>Checks if the floating point type has tinyness before.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>has_signaling_NaN</td>
<td>Checks if the floating point type has signaling NaN.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>radix</td>
<td>Checks the radix of the floating point type.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>traps</td>
<td>Checks if the floating point type traps.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>digits</td>
<td>Checks the number of digits of the floating point type.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>digits10</td>
<td>Checks the number of digits in base 10 of the floating point type.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_exact</td>
<td>Checks if the floating point type is exact.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_iec559</td>
<td>Checks if the floating point type is IEC 559.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_modulo</td>
<td>Checks if the floating point type is modulo.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>is_signed</td>
<td>Checks if the floating point type is signed.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>has_signaling_NaN</td>
<td>Checks if the floating point type has signaling NaN.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>radix</td>
<td>Checks the radix of the floating point type.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>traps</td>
<td>Checks if the floating point type traps.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>digits</td>
<td>Checks the number of digits of the floating point type.</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>digits10</td>
<td>Checks the number of digits in base 10 of the floating point type.</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>
16 Libraries

LSB Core - Generic 5.0

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;wchar_t&gt;::is_exact(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;wchar_t&gt;::is_iec559(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;wchar_t&gt;::is_modulo(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;wchar_t&gt;::is_signed(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.47 struct numeric_limits<unsigned char>

16.1.47.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<unsigned char> specified in Table 16-128, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-128 libstdcxx - struct numeric_limits<unsigned char> Data Interfaces

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::has_denorm(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::is_bounded(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::is_integer(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::round_style(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::has_infinity(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::max_exponent(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::min_exponent(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::has_quiet_NaN(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::is_specialized(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::max_exponent10(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::min_exponent10(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::has_denorm_loss(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::tinyness_before(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::has_signaling_NaN(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::radix(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::traps(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::digits(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;unsigned char&gt;::digits10(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.48 struct numeric_limits<signed char>

16.1.48.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<signed char> specified in Table 16-129, with the full mandatory func-
tionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 16-129 libstdcxx - struct numeric_limits&lt;signed char&gt; Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;signed char&gt;::has_denorm(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::is_bounded(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::is_integer(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::round_style(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::has_infinity(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::max_exponent(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::min_exponent(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::is_specialized(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::max_exponent10(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::min_exponent10(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::tinyness_before(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::radix(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::traps(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::digits(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::digits10(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::is_exact(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::is_iec559(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::is_modulo(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;signed char&gt;::is_signed(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.49 struct numeric_limits<char>

16.1.49.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<char> specified in Table 16-130, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 16-130 libstdcxx - struct numeric_limits&lt;char&gt; Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;char&gt;::has_denorm(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;char&gt;::is_bounded(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;char&gt;::is_integer(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;char&gt;::round_style(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;char&gt;::has_infinity(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;char&gt;::max_exponent(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;char&gt;::min_exponent(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;char&gt;::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;char&gt;::is_specialized(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;char&gt;::max_exponent10(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>numeric_limits&lt;char&gt;::min_exponent10(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>
16 Libraries

16.1.50 struct numeric_limits<bool>

16.1.50.1 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric_limits<bool> specified in Table 16-131, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-131 libstdcxx - struct numeric_limits<bool> Data Interfaces

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Description</th>
<th>Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric_limits&lt;bool&gt;::has_denorm(GLIBCXX_3.4)</td>
<td>has_denorm</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::is_bounded(GLIBCXX_3.4)</td>
<td>is_bounded</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::is_integer(GLIBCXX_3.4)</td>
<td>is_integer</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::round_style(GLIBCXX_3.4)</td>
<td>round_style</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::has_infinity(GLIBCXX_3.4)</td>
<td>has_infinity</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::max_exponent(GLIBCXX_3.4)</td>
<td>max_exponent</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::min_exponent(GLIBCXX_3.4)</td>
<td>min_exponent</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::has_quiet_NaN(GLIBCXX_3.4)</td>
<td>has_quiet_NaN</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::is_specialized(GLIBCXX_3.4)</td>
<td>is_specialized</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::max_exponent10(GLIBCXX_3.4)</td>
<td>max_exponent10</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::min_exponent10(GLIBCXX_3.4)</td>
<td>min_exponent10</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::has_denorm_loss(GLIBCXX_3.4)</td>
<td>has_denorm_loss</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::tinyness_before(GLIBCXX_3.4)</td>
<td>tinyness_before</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::has_signaling_NaN(GLIBCXX_3.4)</td>
<td>has_signaling_NaN</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::radix(GLIBCXX_3.4)</td>
<td>radix</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::traps(GLIBCXX_3.4)</td>
<td>traps</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::digits(GLIBCXX_3.4)</td>
<td>digits</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::digits10(GLIBCXX_3.4)</td>
<td>digits10</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::is_exact(GLIBCXX_3.4)</td>
<td>is_exact</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::is_iec559(GLIBCXX_3.4)</td>
<td>is_iec559</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::is_modulo(GLIBCXX_3.4)</td>
<td>is_modulo</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>numeric_limits&lt;bool&gt;::is_signed(GLIBCXX_3.4)</td>
<td>is_signed</td>
<td>GLIBCXX_3.4</td>
</tr>
</tbody>
</table>
16.1.51 Class ctype_base

16.1.51.1 Class data for ctype_base
The Run Time Type Information for the std::ctype_base class is described by Table 16-132

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for ctype_base</td>
</tr>
</tbody>
</table>

16.1.51.2 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for Class std::ctype_base specified in Table 16-133, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>ctype_base::alnum(GLIBCXX_3.4)</th>
<th>[ISOCXX]</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctype_base::alpha(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>ctype_base::cntrl(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>ctype_base::digit(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>ctype_base::graph(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>ctype_base::lower(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>ctype_base::print(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>ctype_base::punct(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>ctype_base::space(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>ctype_base::upper(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>ctype_base::xdigit(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>typeinfo for ctype_base(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for ctype_base(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.52 Class __ctype_abstract_base<char>

16.1.52.1 Class data for __ctype_abstract_base<char>
The virtual table for the std::__ctype_abstract_base<char> class is described by Table 16-134

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __ctype_abstract_base&lt;char&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>NULL or __ctype_abstract_base&lt;char&gt;::~__ctype_abstract_base()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>NULL or __ctype_abstract_base&lt;char&gt;::~__ctype_abstract_base()</td>
</tr>
</tbody>
</table>
16.1.52.2 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for Class std::__ctype_abstract_base<char> specified in Table 16-135, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 16-135</th>
<th>libstdcxx - Class __ctype_abstract_base&lt;char&gt; Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for</td>
<td>__ctype_abstract_base&lt;char&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name</td>
<td>__ctype_abstract_base&lt;char&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for</td>
<td>__ctype_abstract_base&lt;char&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.53 Class __ctype_abstract_base<wchar_t>

16.1.53.1 Class data for __ctype_abstract_base<wchar_t>

The virtual table for the std::__ctype_abstract_base<wchar_t> class is described by Table 16-136

<table>
<thead>
<tr>
<th>Table 16-136</th>
<th>Primary vtable for __ctype_abstract_base&lt;wchar_t&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __ctype_abstract_base&lt;wchar_t&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>NULL or __ctype_abstract_base&lt;wchar_t&gt;::~__ctype_abstract_base()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>NULL or __ctype_abstract_base&lt;wchar_t&gt;::~__ctype_abstract_base()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>__cxa_pure_virtual</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>__cxa_pure_virtual</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>__cxa_pure_virtual</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>__cxa_pure_virtual</td>
</tr>
</tbody>
</table>
16.1.53.2 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for Class std::__ctype_abstract_base<wchar_t> specified in Table 16-137, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-137 libstdcxx - Class __ctype_abstract_base<wchar_t> Data Interfaces

<table>
<thead>
<tr>
<th>Typeinfo for __ctype_abstract_base&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typeinfo name for __ctype_abstract_base&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>Vtable for __ctype_abstract_base&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.54 Class ctype<char>

16.1.54.1 Class data for ctype<char>

The virtual table for the std::ctype<char> class is described by Table 16-138.

Table 16-138 Primary vtable for ctype<char>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for ctype&lt;char&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>ctype&lt;char&gt;::~ctype()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>ctype&lt;char&gt;::~ctype()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>ctype&lt;char&gt;::do_toupper(char) const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>ctype&lt;char&gt;::do_toupper(char*, char const*) const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>ctype&lt;char&gt;::do_tolower(char) const</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>ctype&lt;char&gt;::do_tolower(char*, char const*) const</td>
</tr>
<tr>
<td>vfunc[6]:</td>
<td>ctype&lt;char&gt;::do_widen(char) const</td>
</tr>
<tr>
<td>vfunc[7]:</td>
<td>ctype&lt;char&gt;::do_widen(char const*, char const*, char*) const</td>
</tr>
<tr>
<td>vfunc[8]:</td>
<td>ctype&lt;char&gt;::do_narrow(char, char) const</td>
</tr>
<tr>
<td>vfunc[9]:</td>
<td>ctype&lt;char&gt;::do_narrow(char const*, char const*, char, char*) const</td>
</tr>
</tbody>
</table>
16.1.54.2 Interfaces for Class `ctype<char>`

An LSB conforming implementation shall provide the generic methods for Class `std::ctype<char>` specified in Table 16-139, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>do_tolower(char*, char const*) const</code></td>
<td>Convert characters to lowercase</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>do_tolower(char) const</code></td>
<td>Convert a single character to lowercase</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>do_toupper(char*, char const*) const</code></td>
<td>Convert characters to uppercase</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>do_toupper(char) const</code></td>
<td>Convert a single character to uppercase</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>do_widen(char const*, char const*, char*) const</code></td>
<td>Expand characters to wider type</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>do_widen(char) const</code></td>
<td>Expand a single character to a wider type</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>do_narrow(char const*, char const*, char, char*) const</code></td>
<td>Narrow characters to narrower type</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>do_narrow(char, char) const</code></td>
<td>Narrow a single character to a narrower type</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>classic_table()</code></td>
<td>Get the default character table</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>~ctype()</code></td>
<td>Destructor</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>~ctype()</code></td>
<td>Destructor</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>~ctype()</code></td>
<td>Destructor</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>has_facet&lt;ctype&lt;char&gt;&gt;(locale const&amp;)</code></td>
<td>Check for facet availability</td>
<td>GLIBCXX_3.4</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::ctype<char>` specified in Table 16-140, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Data Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>table_size()</code></td>
<td>Get the table size</td>
</tr>
<tr>
<td><code>id()</code></td>
<td>Get the class identifier</td>
</tr>
<tr>
<td><code>typeinfo</code></td>
<td>Get the typeinfo</td>
</tr>
<tr>
<td><code>name</code></td>
<td>Get the type name</td>
</tr>
<tr>
<td><code>vtable</code></td>
<td>Get the virtual table</td>
</tr>
</tbody>
</table>

16.1.55 Class `ctype<wchar_t>`

16.1.55.1 Class data for `ctype<wchar_t>`

The virtual table for the `std::ctype<wchar_t>` class is described by Table 16-141.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for <code>ctype&lt;wchar_t&gt;</code></td>
</tr>
<tr>
<td>vfunc[0]</td>
<td><code>ctype&lt;wchar_t&gt;::~ctype()</code></td>
</tr>
<tr>
<td>vfunc[1]</td>
<td><code>ctype&lt;wchar_t&gt;::~ctype()</code></td>
</tr>
<tr>
<td>vfunc[2]</td>
<td><code>ctype&lt;wchar_t&gt;::do_is(unsigned short, wchar_t const)</code></td>
</tr>
<tr>
<td>vfunc[3]</td>
<td><code>ctype&lt;wchar_t&gt;::do_is(wchar_t const*, wchar_t const*, unsigned short*) const</code></td>
</tr>
</tbody>
</table>
vfunc[4]:  ctype<wchar_t>::do_scan_is(unsigned short, wchar_t const*, wchar_t const*) const
vfunc[5]:  ctype<wchar_t>::do_scan_not(unsigned short, wchar_t const*, wchar_t const*) const
vfunc[6]:  ctype<wchar_t>::do_toupper(wchar_t) const
vfunc[7]:  ctype<wchar_t>::do_toupper(wchar_t*, wchar_t const*) const
vfunc[8]:  ctype<wchar_t>::do_tolower(wchar_t) const
vfunc[9]:  ctype<wchar_t>::do_tolower(wchar_t*, wchar_t const*) const
vfunc[10]: ctype<wchar_t>::do_widen(char) const
vfunc[11]: ctype<wchar_t>::do_widen(char const*, char const*, wchar_t*) const
vfunc[12]: ctype<wchar_t>::do_narrow(wchar_t, char) const
vfunc[13]: ctype<wchar_t>::do_narrow(wchar_t const*, wchar_t const*, char, char*) const

The Run Time Type Information for the std::ctype<wchar_t> class is described by Table 16-142.

### Table 16-142 typeinfo for ctype<wchar_t>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for ctype&lt;wchar_t&gt;</td>
</tr>
</tbody>
</table>

#### 16.1.55.2 Interfaces for Class ctype<wchar_t>

An LSB conforming implementation shall provide the generic methods for Class std::ctype<wchar_t> specified in Table 16-143, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-143 libstdcxx - Class ctype<wchar_t> Function Interfaces

| ctype<wchar_t>::do_scan_is(unsigned short, wchar_t const*, wchar_t const*) const(GLIBCXX_3.4) [ISOCXX] |
| ctype<wchar_t>::do_tolower(wchar_t*, wchar_t const*) const(GLIBCXX_3.4) [ISOCXX] |
| ctype<wchar_t>::do_narrow(wchar_t, char) const(GLIBCXX_3.4) [ISOCXX] |
| ctype<wchar_t>::do_narrow(wchar_t const*, wchar_t const*, char, char*) const(GLIBCXX_3.4) [ISOCXX] |
| ctype<wchar_t>::do_scan_not(unsigned short, wchar_t const*, wchar_t const*) const(GLIBCXX_3.4) [ISOCXX] |
| ctype<wchar_t>::do_is(wchar_t const*, wchar_t const*, unsigned short*) const(GLIBCXX_3.4) [ISOCXX] |
An LSB conforming implementation shall provide the generic data interfaces for Class std::ctype<wchar_t> specified in Table 16-144, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-144 libstdcxx - Class ctype<wchar_t> Data Interfaces

<table>
<thead>
<tr>
<th>Method</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctype&lt;wchar_t&gt;::id(GLIBCXX_3.4) [ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>typeinfo for ctype&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
<td></td>
</tr>
<tr>
<td>typeinfo name for ctype&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
<td></td>
</tr>
<tr>
<td>vtable for ctype&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
<td></td>
</tr>
</tbody>
</table>

### 16.1.56 Class ctypebyname<char>

#### 16.1.56.1 Class data for ctypebyname<char>

The virtual table for the std::ctypebyname<char> class is described by Table 16-145.

### Table 16-145 Primary vtable for ctypebyname<char>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>Vfunc 0:</th>
<th>Vfunc 1:</th>
<th>Vfunc 2:</th>
<th>Vfunc 3:</th>
<th>Vfunc 4:</th>
<th>Vfunc 5:</th>
<th>Vfunc 6:</th>
<th>Vfunc 7:</th>
<th>Vfunc 8:</th>
<th>Vfunc 9:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RTTI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::ctypebyname<char> class is described by Table 16-146.
16.1.56.2 Interfaces for Class `ctype_byname<char>`

An LSB conforming implementation shall provide the generic methods for Class `std::ctype_byname<char>` specified in Table 16-147, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Type</th>
<th>LSBCYY_3.4</th>
<th>ISOCXX</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ctype_byname&lt;char&gt;::~ctype_byname()</code> (GLIBCXX_3.4)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>ctype_byname&lt;char&gt;::~ctype_byname()</code> (GLIBCXX_3.4)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>ctype_byname&lt;char&gt;::~ctype_byname()</code> (GLIBCXX_3.4)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::ctype_byname<char>` specified in Table 16-148, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Interface</th>
<th>Type</th>
<th>LSBCYY_3.4</th>
<th>ISOCXX</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo name for <code>ctype_byname&lt;char&gt;</code> (GLIBCXX_3.4)</td>
<td></td>
<td>[CXXABI-1.86]</td>
<td></td>
</tr>
<tr>
<td>vtable name for <code>ctype_byname&lt;char&gt;</code> (GLIBCXX_3.4)</td>
<td></td>
<td>[CXXABI-1.86]</td>
<td></td>
</tr>
<tr>
<td>vtable for <code>ctype_byname&lt;char&gt;</code> (GLIBCXX_3.4)</td>
<td></td>
<td>[CXXABI-1.86]</td>
<td></td>
</tr>
</tbody>
</table>

16.1.57 Class `ctype_byname<wchar_t>`

16.1.57.1 Class data for `ctype_byname<wchar_t>`

The virtual table for the std::ctype_byname<wchar_t> class is described by Table 16-149.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Typeinfo name for <code>ctype_byname&lt;wchar_t&gt;</code></th>
<th>RTTI</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>typeinfo name for <code>ctype_byname&lt;wchar_t&gt;</code></td>
<td>vfunc[0]:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>vfunc[1]:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>vfunc[2]:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>vfunc[3]:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>vfunc[4]:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>vfunc[5]:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>vfunc[6]:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>vfunc[7]:</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the std::ctype_byname<wchar_t> class is described by Table 16-150.

Table 16-150 typeinfo for ctype_byname<wchar_t>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for ctype_byname&lt;wchar_t&gt;</td>
</tr>
</tbody>
</table>

16.1.57.2 Interfaces for Class ctype_byname<wchar_t>

An LSB conforming implementation shall provide the generic methods for Class std::ctype_byname<wchar_t> specified in Table 16-151, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-151 libstdcxx - Class ctype_byname<wchar_t> Function Interfaces

<table>
<thead>
<tr>
<th>ctype_byname&lt;wchar_t&gt;::~ctype_byname() {GLIBCXX_3.4} [ISOCXX]</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctype_byname&lt;wchar_t&gt;::~ctype_byname() {GLIBCXX_3.4} [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::ctype_byname<wchar_t> specified in Table 16-152, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-152 libstdcxx - Class ctype_byname<wchar_t> Data Interfaces

<table>
<thead>
<tr>
<th>typeinfo for ctype_byname&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo name for ctype_byname&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for ctype_byname&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.58 Class basic_string<char, char_traits<char>, allocator<char> >

16.1.58.1 Interfaces for Class basic_string<char, char_traits<char>, allocator<char> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_string<char, std::char_traits<char>, std::allocator<char> > specified in Table 16-153, with the full mandatory functionality as described in the referenced underlying specification.
<table>
<thead>
<tr>
<th>Function Name</th>
<th>GLIBCXX Version</th>
<th>ISOCXX Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_M_disjunct(char const*)</td>
<td>GLIBCXX 3.4.5</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::get_allocator()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::end() const</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Rep::_M_is_leaked()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Rep::_M_is_shared()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::data()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::rend()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::size()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::begin()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::c_str()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::empty()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_M_rep()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::length()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::rbegin()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_M_data()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_M_iend()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::compare(char const*)</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::compare(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::capacity()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::max_size()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_M_ibegin()</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Alloc_hider::_Alloc_hider(char*, allocator&lt;char&gt; const&amp;)(GLIBCXX 3.4)</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Alloc_hider::_Alloc_hider(char*, allocator&lt;char&gt; const&amp;)(GLIBCXX 3.4)</td>
<td>GLIBCXX 3.4</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>
basic_string<char, char_traits<char>, allocator<char> >::_M_leak_hard()
  (GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::_S_empty_rep()
  (GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::_S_copy_chars(char*,
  __gnu_cxx::__normal_iterator<char const*, basic_string<char, char_traits<char>,
  allocator<char> > >, __gnu_cxx::__normal_iterator<char const*, basic_string<char,
  char_traits<char>, allocator<char> > >)(GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::_S_copy_chars(char*,
  __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> > >)(GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::_S_copy_chars(char*,
  char const*, char const*)(GLIBCXX_3.4)

basic_string<char, char_traits<char>, allocator<char> >::_S_copy_chars(char*,
  char*, char*)(GLIBCXX_3.4)

basic_string<char, char_traits<char>, allocator<char> >::end()

basic_string<char, char_traits<char>, allocator<char> >::_Rep::_M_destroy(allocator<char> const&)
  (GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::_Rep::_M_dispose(allocator<char> const&)
  (GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::_Rep::_M_refcopy()
  (GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::_Rep::_M_refdata()
  (GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::_Rep::_S_empty_rep()
  (GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::_Rep::_M_set_leaked()
  (GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::_Rep::_M_set_sharable()
  (GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::_Rep::_M_grab(allocator<char> const&,
  allocator<char> const&)
  (GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::rend()

basic_string<char, char_traits<char>, allocator<char> >::swap(basic_string<char,
  char_traits<char>, allocator<char> >&)(GLIBCXX_3.4)

basic_string<char, char_traits<char>, allocator<char> >::begin()

basic_string<char, char_traits<char>, allocator<char> >::clear()

basic_string<char, char_traits<char>, allocator<char> >::erase(__gnu_cxx::__normal_iterator<char*,
  basic_string<char, char_traits<char>, allocator<char> > > >)
  (GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::erase(__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> > >)(GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::append(char const*)
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::append(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4)</code></td>
<td>Adds the contents of another string to the end of the current string.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::assign(char const*) (GLIBCXX_3.4)</code></td>
<td>Assigns a null-terminated character string to the current string.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::assign(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4)</code></td>
<td>Assigns the contents of another string to the current string.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::insert(__gnu_cxx::__normal_iterator&lt;char*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; &gt;, char)(GLIBCXX_3.4)</code></td>
<td>Inserts a character into the string at the specified position.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::rbegin()(GLIBCXX_3.4)</code></td>
<td>Returns a reverse iterator to the end of the string.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_M_data(char*) (GLIBCXX_3.4)</code></td>
<td>Accesses the memory to which the string is allocated.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_M_leak()(GLIBCXX_3.4)</code></td>
<td>Releases the memory allocation for the string.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::replace(__gnu_cxx::__normal_iterator&lt;char*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;char*&gt;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4)</code></td>
<td>Replaces the characters between two iterators with the characters of another string.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::replace(__gnu_cxx::__normal_iterator&lt;char*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;char*&gt;, __gnu_cxx::__normal_iterator&lt;char const*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;char const*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; &gt;)(GLIBCXX_3.4)</code></td>
<td>Replaces the characters between two iterators with the characters of another string.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::replace(__gnu_cxx::__normal_iterator&lt;char*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;char*&gt;, char const*) (GLIBCXX_3.4)</code></td>
<td>Replaces the characters between two iterators with a null-terminated character string.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::replace(__gnu_cxx::__normal_iterator&lt;char*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;char*&gt;, char const*, char const*) (GLIBCXX_3.4)</code></td>
<td>Replaces the characters between two iterators with a null-terminated character string.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::push_back(char)</code> (GLIBCXX_3.4)`</td>
<td>Adds a single character to the end of the string.</td>
</tr>
<tr>
<td><code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_string(char const*) (GLIBCXX_3.4)</code></td>
<td>Constructs a string from a null-terminated character string.</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
allocator<
char> const&)(GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::basic_string<
allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::basic_string(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&)(GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::basic_string(char const*, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::basic_string(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&) (GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::basic_string<char const*>(char const*, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::basic_string<char*>(char*, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::operator=(char const*) (GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::operator=(basic_string<
char, char_traits<char>, allocator<char> > const&) (GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::operator=(char) (GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::operator+=(char const*) (GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::~basic_string() (GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::~basic_string() (GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::operator=(char const*) (GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::operator=(char) (GLIBCXX_3.4) [ISOCXX]

basic_string<
char, char_traits<char>, allocator<char>
>::operator+=(char const*) (GLIBCXX_3.4) [ISOCXX]
An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_string<char, std::char_traits<char>, std::allocator<char>> specified in Table 16-154, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-154  libstdcxx - Class std::basic_string<char, std::char_traits<char>, std::allocator<char>> Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&amp; operator+(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&amp; l, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&amp; r)</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&amp; operator+(char c)</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.59 Class std::basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

16.1.59.1 Interfaces for Class std::basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

An LSB conforming implementation shall provide the generic methods for Class std::basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> specified in Table 16-155, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-155  libstdcxx - Class std::basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;&amp; _M_disjunct(wchar_t const*) const</td>
<td>GLIBCXX_3.4.5 [ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;&amp; get_allocator() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;&amp; end()</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;&amp; _M_is_leaked() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;&amp; _M_is_shared() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;&amp; data()</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;&amp; rend()</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;&amp; size()</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;&amp; begin()</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
<table>
<thead>
<tr>
<th>Member Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::c_str()</code></td>
</tr>
<tr>
<td><code>const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::empty()</code></td>
</tr>
<tr>
<td><code>const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::_M_rep()</code></td>
</tr>
<tr>
<td><code>const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::length()</code></td>
</tr>
<tr>
<td><code>const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::rbegin()</code></td>
</tr>
<tr>
<td><code>const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::_M_data()</code></td>
</tr>
<tr>
<td><code>const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::_M_iend()</code></td>
</tr>
<tr>
<td><code>const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::compare(wchar_t const*) const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::compare(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;) const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::capacity()</code></td>
</tr>
<tr>
<td><code>const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::max_size()</code></td>
</tr>
<tr>
<td><code>const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::_M_ibegin()</code></td>
</tr>
<tr>
<td><code>const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::_Alloc_hider::_Alloc_hider(wchar_t*, allocator&lt;wchar_t&gt; const&amp;)</code></td>
</tr>
<tr>
<td><code>GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::_S_empty_rep()</code></td>
</tr>
<tr>
<td><code>GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::_S_copy_chars(wchar_t*, __gnu_cxx::__normal_iterator&lt;wchar_t const*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;)</code></td>
</tr>
<tr>
<td><code>GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::_S_copy_chars(wchar_t*, wchar_t const*, wchar_t const*)</code></td>
</tr>
<tr>
<td><code>GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::_S_copy_chars(wchar_t*, wchar_t*, wchar_t*)</code></td>
</tr>
<tr>
<td><code>GLIBCXX_3.4)</code></td>
</tr>
</tbody>
</table>
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::end() (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::M_destroy(allocator<wchar_t> const&) (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::_M_destroy(allocator<wchar_t> const&) (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::_M_dispose(allocator<wchar_t> const&) (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::_M_refcopy() (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::_M_refdata() (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::S_empty_rep() (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::M_set_leaked() (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::M_set_sharable() (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::M_grab(allocator<wchar_t> const&, allocator<wchar_t> const&) (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::rend() (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::swap(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >&) (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::begin() (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::clear() (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::erase(__gnu_cxx::__normal_iterator<wchar_t*, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > >) (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::erase(__gnu_cxx::__normal_iterator<wchar_t*, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > >, __gnu_cxx::__normal_iterator<wchar_t*, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > >) (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::append(wchar_t const*) (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&) (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::assign(wchar_t const*) (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&) (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::insert(__gnu_cxx::__normal_iterator<wchar_t*, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > >, wchar_t) (GLIBCXX_3.4) [ISOcxx]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::rbegin() (GLIBCXX_3.4) [ISOcxx]

© 2015 Linux Foundation
<table>
<thead>
<tr>
<th>Function</th>
<th>Library</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::_M_data(wchar_t*)</td>
<td>GLIBCXX_3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::_M_leak()</td>
<td>GLIBCXX_3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::replace(__gnu_cxx::__normal_iterator&lt;wchar_t*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;wchar_t*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;wchar_t const*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;wchar_t const*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::replace(__gnu_cxx::__normal_iterator&lt;wchar_t*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;wchar_t*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;, wchar_t const*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::replace(__gnu_cxx::__normal_iterator&lt;wchar_t*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;wchar_t*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::replace(__gnu_cxx::__normal_iterator&lt;wchar_t*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;wchar_t*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;wchar_t*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;, __gnu_cxx::__normal_iterator&lt;wchar_t*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; &gt;)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::push_back(wchar_t)</td>
<td>GLIBCXX_3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_string(wchar_t const*, allocator&lt;wchar_t&gt; const&amp;)</td>
<td>GLIBCXX_3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_string(allocator&lt;wchar_t&gt; const&amp;)</td>
<td>GLIBCXX_3.4</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_string(wchar_t const*, allocator&lt;wchar_t&gt; const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_string(wchar_t const*, allocator&lt;wchar_t&gt; const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
<td></td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_string(wchar_t const*, allocator&lt;wchar_t&gt; const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
<td></td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class
std::basic_string<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > specified in Table 16-156, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-156 libstdcxx - Class basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Declaration</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::npos(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::npos(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::npos(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::npos(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>
16.1.60 Class basic_stringstream<char, char_traits<char>, allocator<char> >

16.1.60.1 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 in the relevant architecture specific part of this specification.

The VTT for the std::basic_stringstream<char, std::char_traits<char>, std::allocator<char> > class is described by Table 16-157.

Table 16-157 VTT for basic_stringstream<char, char_traits<char>, allocator<char> >

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>Number of Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ZTTSt18basic_stringstreamIcSt11char_traitsIcESaIcEE</td>
<td>10</td>
</tr>
</tbody>
</table>

16.1.60.2 Interfaces for Class basic_stringstream<char, char_traits<char>, allocator<char> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_stringstream<char, std::char_traits<char>, std::allocator<char> > specified in Table 16-158, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-158 libstdcxx - Class basic_stringstream<char, char_traits<char>, allocator<char> > Function Interfaces

<table>
<thead>
<tr>
<th>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::str() const(GLIBCXX_3.4) [ISOCXX]</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::dbuf() const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::str(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;) (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_stringstream(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_stringstream(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::~basic_stringstream()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::~basic_stringstream()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::~basic_stringstream()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_stringstream<char, std::char_traits<char>, std::allocator<char> > specified in
Table 16.159, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-159  libstdcxx - Class basic_stringstream<char, char_traits<char>, allocator<char> >

<table>
<thead>
<tr>
<th>Type Information</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typename</td>
<td>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable</td>
<td>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.61 Class basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

16.1.61.1 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 in the relevant architecture specific part of this specification.

The VTT for the std::basic_stringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > class is described by Table 16-160.

Table 16-160  VTT for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ZTTS18basic_stringstreamIwSt11char_traitsIwESaIwEE</td>
<td>Number of Entries 10</td>
</tr>
</tbody>
</table>

16.1.61.2 Interfaces for Class basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_stringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > specified in Table 16-161, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-161  libstdcxx - Class basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::str()</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::rdbuf()</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::str(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;)</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::str(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp; _Ios_Openmode)</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::rdbuf(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp; _Ios_Openmode)</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class
\(\text{std::basic_stringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>}\)
specified in
Table 16-162, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>Number of Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ZTTSt19basic_stringstreamIcSt11char_traitsIcESaIcEE</td>
<td>4</td>
</tr>
</tbody>
</table>

16.1.62.2 Interfaces for Class basic_istringstream<char, char_traits<char>, allocator<char>>

An LSB conforming implementation shall provide the generic methods for Class
\(\text{std::basic_istringstream<char, std::char_traits<char>, std::allocator<char>}\)
specified in
Table 16-164, with the full mandatory functionality as described in the referenced underlying specification.
An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_istringstream<char, std::char_traits<char>, std::allocator<char> > specified in Table 16-165, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-165 libstdcxx - Class basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

Table 16-166 Class basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

16.1.63 Class basic_istringstream<br>char_traits<char>, allocator<char> >

16.1.63.1 Class data for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

The virtual table for the std::basic_istringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > class is described in the relevant architecture specific part of this specification.

The VTT for the std::basic_istringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > class is described by Table 16-166.
Table 16-166 VTT for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>_ZTTSt19basic_istringstreamIwSt11char_traitsIwESaIwEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Entries</td>
<td>4</td>
</tr>
</tbody>
</table>

16.1.63.2 Interfaces for Class basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_istringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > specified in Table 16-167, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-167 libstdcxx - Class basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > Function Interfaces

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_istringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::str() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_istringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::rdbuf() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_istringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::str(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;) (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_istringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_istringstream(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;, _Ios_Openmode) (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_istringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_istringstream(_Ios_Openmode) (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_istringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~basic_istringstream() (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_istringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~basic_istringstream() (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_istringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~basic_istringstream() (GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_istringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > specified in Table 16-168, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-168 libstdcxx - Class basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > Data Interfaces

<table>
<thead>
<tr>
<th>Data Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for basic_istringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; (GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for basic_istringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; (GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>VTT for basic_istringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; (GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
16 Libraries

LSB Core - Generic 5.0

vttable for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >(GLIBCXX_3.4) [CXXABI-1.86]

16.1.64 Class basic_ostringstream<char, char_traits<char>, allocator<char> >

16.1.64.1 Class data for basic_ostringstream<char, char_traits<char>, allocator<char> >

The virtual table for the std::basic_ostringstream<char, std::char_traits<char>, std::allocator<char> > class is described in the relevant architecture specific part of this specification.

The VTT for the std::basic_ostringstream<char, std::char_traits<char>, std::allocator<char> > class is described by Table 16-169.

Table 16-169 VTT for basic_ostringstream<char, char_traits<char>, allocator<char> >

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>_ZTTSt19basic_ostringstreamIcSt11char_traitsIcESaIcEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Entries</td>
<td>4</td>
</tr>
</tbody>
</table>

16.1.64.2 Interfaces for Class basic_ostringstream<char, char_traits<char>, allocator<char> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_ostringstream<char, std::char_traits<char>, std::allocator<char> > specified in Table 16-170, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-170 libstdcxx - Class basic_ostringstream<char, char_traits<char>, allocator<char> > Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_ostringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::str()</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ostringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::rdbuf()</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ostringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::str(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ostringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_ostringstream(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;, _Ios_Openmode)</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ostringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_ostringstream(_Ios_Openmode)</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ostringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::~basic_ostringstream()</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_ostream<char, std::char_traits<char>, std::allocator<char>> > specified in Table 16-171, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-171 libstdcxx - Class basic_ostream<char, char_traits<char>, allocator<char>> > Data Interfaces

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>Number of Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeid for basic_ostream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt; (GLIBCXX_3.4) [CXXABI-1.86]</td>
<td></td>
</tr>
<tr>
<td>typename name for basic_ostream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt; (GLIBCXX_3.4) [CXXABI-1.86]</td>
<td></td>
</tr>
<tr>
<td>VTT for basic_ostream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt; (GLIBCXX_3.4) [CXXABI-1.86]</td>
<td></td>
</tr>
<tr>
<td>vtable for basic_ostream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt; (GLIBCXX_3.4) [CXXABI-1.86]</td>
<td></td>
</tr>
</tbody>
</table>

16.1.65 Class basic_ostream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >

16.1.65.1 Class data for basic_ostream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >

The virtual table for the std::basic_ostream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> > class is described in the relevant architecture specific part of this specification.

The VTT for the std::basic_ostream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> > class is described by Table 16-172.

Table 16-172 VTT for basic_ostream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>Number of Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ZTTSt19basic_ostreamIwSt11char_traitsIwESaIwEE</td>
<td>4</td>
</tr>
</tbody>
</table>

16.1.65.2 Interfaces for Class basic_ostream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_ostream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> > specified in Table 16-173, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-173 libstdcxx - Class basic_ostream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> > Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt; ::str(const(GLIBCXX_3.4) [ISOCXX])</td>
</tr>
<tr>
<td>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt; ::dbuf(const(GLIBCXX_3.4) [ISOCXX])</td>
</tr>
<tr>
<td>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt; ::str(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt; const&amp;) (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt; ::basic_ostream(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt; const&amp;, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt;</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
An LSB conforming implementation shall provide the generic data interfaces for Class
std::basic_ostringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> >
specified in Table 16-174, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-174  libstdcxx - Class basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > > Data Interfaces

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeid</td>
<td>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;</td>
</tr>
<tr>
<td>vfunc[0]</td>
<td>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~</td>
</tr>
<tr>
<td>vfunc[1]</td>
<td>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~</td>
</tr>
<tr>
<td>vfunc[2]</td>
<td>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~</td>
</tr>
<tr>
<td>vfunc[3]</td>
<td>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~</td>
</tr>
<tr>
<td>vfunc[4]</td>
<td>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~</td>
</tr>
</tbody>
</table>

16.1.66 Class basic_stringbuf<char, char_traits<char>, allocator<char> >

16.1.66.1 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 16-175.

Table 16-175  Primary vtable for basic_stringbuf<char, char_traits<char>, allocator<char> >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeid for basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;</td>
</tr>
<tr>
<td>vfunc[0]</td>
<td>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::~basic_stringbuf()</td>
</tr>
<tr>
<td>vfunc[1]</td>
<td>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::~basic_stringbuf()</td>
</tr>
<tr>
<td>vfunc[2]</td>
<td>basic_streambuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::imbue(locale const&amp;)</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic methods for Class std::basic_stringbuf<char, std::char_traits<char>, std::allocator<char>> > specified in Table 16-177, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-177 libstdcxx - Class basic_stringbuf<char, char_traits<char>, allocator<char>> > Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::str()</td>
<td>Constructor for the stringbuf object. The constructor takes a const reference to a string as its argument.</td>
</tr>
<tr>
<td>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::M_update_egptr()</td>
<td>Updates the end of the buffer pointer to point to the end of the string.</td>
</tr>
<tr>
<td>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::M_stringbuf_init(Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</td>
<td>Initializes the stringbuf object with the specified open mode.</td>
</tr>
<tr>
<td>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::str(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt; &amp;str)</td>
<td>Overloads the string copy constructor.</td>
</tr>
<tr>
<td>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::seekpos(fpos&lt;_mbstate_t&gt;, Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</td>
<td>Sets the current position of the stringbuf to the specified position.</td>
</tr>
<tr>
<td>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::overflow(int)</td>
<td>Overloads the overflow operator.</td>
</tr>
<tr>
<td>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::pbackfail(int)</td>
<td>Overloads the pbackfail operator.</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::basic_stringbuf<char, std::char_traits<char>, std::allocator<char>> > class is described by Table 16-176.

### Table 16-176 typeinfo for basic_stringbuf<char, char_traits<char>, allocator<char>> >

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;</td>
</tr>
</tbody>
</table>

16.1.66.2 Interfaces for Class basic_stringbuf<char, char_traits<char>, allocator<char>> >
An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_stringbuf<char, std::char_traits<char>, std::allocator<char> > specified in Table 16-178, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-178 libstdcxx - Class basic_stringbuf<char, char_traits<char>, allocator<char> > | Data Interfaces

<table>
<thead>
<tr>
<th>Virtual Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTTI</td>
<td>typeinfo for basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;</td>
</tr>
<tr>
<td>vtable for basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;</td>
<td>vfunc[0]: basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::~basic_stringbuf()</td>
</tr>
</tbody>
</table>

16.1.67 Class basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

16.1.67.1 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 16-179.

Table 16-179 Primary vtable for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::~basic_stringbuf()</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the std::basic_stringbuf<wchar_t, \nstd::char_traits<wchar_t>, std::allocator<wchar_t>> class is described by Table 16-180

Table 16-180 typeinfo for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for _cxxabiv1:: _si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for basic_stringbuf&lt;wchar_t,\nchar_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</td>
</tr>
</tbody>
</table>

### 16.1.67.2 Interfaces for Class basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>

An LSB conforming implementation shall provide the generic methods for Class std::basic_stringbuf<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>> as specified in Table 16-181, with the full mandatory functionality as described in the referenced underlying specification.
### Table 16-181 libstdcxx - Class `basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>` > Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>str()</code></td>
<td>Constant function that returns a reference to the stringbuf object.</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>M_update_egptr()</code></td>
<td>Function that updates the end of string pointer.</td>
<td>ISOCXX</td>
</tr>
<tr>
<td><code>M_stringbuf_init(_Ios_Openmode)</code></td>
<td>Constructor that initializes the basic_stringbuf object.</td>
<td>ISOCXX</td>
</tr>
<tr>
<td><code>str(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;)</code></td>
<td>Function that returns a reference to the stringbuf object.</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>seekpos(fpos&lt;__mbstate_t&gt;, _Ios_Openmode)</code></td>
<td>Function that sets the position in the stringbuf to the specified offset.</td>
<td>ISOCXX</td>
</tr>
<tr>
<td><code>overflow(unsigned int)</code></td>
<td>Function that performs an overflow operation.</td>
<td>ISOCXX</td>
</tr>
<tr>
<td><code>pbackfail(unsigned int)</code></td>
<td>Function that performs a pbackfail operation.</td>
<td>ISOCXX</td>
</tr>
<tr>
<td><code>showmanyc()</code></td>
<td>Function that returns true if the stringbuf object can show more characters.</td>
<td>ISOCXX</td>
</tr>
<tr>
<td><code>underflow()</code></td>
<td>Function that performs an underflow operation.</td>
<td>ISOCXX</td>
</tr>
<tr>
<td><code>basic_stringbuf(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;, _Ios_Openmode)</code></td>
<td>Constructor that initializes the basic_stringbuf object.</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>basic_stringbuf(_Ios_Openmode)</code></td>
<td>Constructor that initializes the basic_stringbuf object.</td>
<td>ISOCXX</td>
</tr>
<tr>
<td><code>basic_stringbuf(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;, _Ios_Openmode)</code></td>
<td>Constructor that initializes the basic_stringbuf object.</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td><code>~basic_stringbuf()</code></td>
<td>Destructor function that deallocates the stringbuf object.</td>
<td>ISOCXX</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_stringbuf<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>>` specified in Table 16-182, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-182 libstdcxx - Class `basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>` > Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>typeinfo for basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code></td>
<td>Function that returns the typeinfo object for the stringbuf class.</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code></td>
<td>Function that returns the typeinfo name for the stringbuf class.</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td><code>vtable for basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code></td>
<td>Function that returns the vtable for the stringbuf class.</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
</tbody>
</table>
16.1.68 Class `basic_iostream<char, char_traits<char>>` 

16.1.68.1 Class data for `basic_iostream<char, char_traits<char>>` 

The virtual table for the `std::basic_iostream<char, std::char_traits<char>>` class is described in the relevant architecture specific part of this specification. 

The VTT for the `std::basic_iostream<char, std::char_traits<char>>` class is described by Table 16-183

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>_ZTTSd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Entries</td>
<td>7</td>
</tr>
</tbody>
</table>

16.1.68.2 Interfaces for Class `basic_iostream<char, char_traits<char>>` 

An LSB conforming implementation shall provide the generic methods for Class `std::basic_iostream<char, std::char_traits<char>>` specified in Table 16-184, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_iostream&lt;char, char_traits&lt;char&gt;&gt;::basic_iostream(basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;*)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_iostream&lt;char, char_traits&lt;char&gt;&gt;::basic_iostream()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_iostream&lt;char, char_traits&lt;char&gt;&gt;::~basic_iostream(basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;*)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_iostream&lt;char, char_traits&lt;char&gt;&gt;::~basic_iostream()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_iostream&lt;char, char_traits&lt;char&gt;&gt;::~basic_iostream()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;::operator&gt;&gt;char_traits&lt;char&gt;(basic_istream&lt;char, char_traits&lt;char&gt;&gt;, signed char*)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_iostream<char, std::char_traits<char>>` specified in Table 16-185, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for basic_iostream&lt;char, char_traits&lt;char&gt;&gt;&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for basic_iostream&lt;char, char_traits&lt;char&gt;&gt;&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>VTT for basic_iostream&lt;char, char_traits&lt;char&gt;&gt;&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for basic_iostream&lt;char, char_traits&lt;char&gt;&gt;&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>
16.1.69 Class basic_iostream<wchar_t, char_traits<wchar_t> >

16.1.69.1 Class data for basic_iostream<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_iostream<wchar_t, std::char_traits<wchar_t> > class is described in the relevant architecture specific part of this specification.

The VTT for the std::basic_iostream<wchar_t, std::char_traits<wchar_t> > class is described by Table 16-186

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>_ZTTSt14basic_iostreamIwSt11char_traitsIwEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Entries</td>
<td>7</td>
</tr>
</tbody>
</table>

16.1.69.2 Interfaces for Class basic_iostream<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_iostream<wchar_t, std::char_traits<wchar_t> > specified in Table 16-187, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_iostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_iostream(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;*) (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_iostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_iostream() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_iostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_iostream(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;*) (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_iostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_iostream() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_iostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_iostream() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_iostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_iostream() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_iostream<wchar_t, std::char_traits<wchar_t> > specified in Table 16-188, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for basic_iostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for basic_iostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>
16.1.70 Class basic_istream<char, char_traits<char> >

16.1.70.1 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 in the relevant architecture specific part of this specification.

The VTT for the std::basic_istream<char, std::char_traits<char> > class is described by Table 16-189

Table 16-189 VTT for basic_istream<char, char_traits<char> >

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>Number of Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ZTTSi</td>
<td>2</td>
</tr>
</tbody>
</table>

16.1.70.2 Interfaces for Class basic_istream<char, char_traits<char> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_istream<char, std::char_traits<char> > specified in Table 16-190, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-190 libstdcxx - Class basic_istream<char, char_traits<char> > Function Interfaces

```cpp
basic_istream<char, char_traits<char> >&::gcount() const(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::sentry::operator bool() const(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::get(basic_streambuf<char, char_traits<char> >&)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::get(basic_streambuf<char, char_traits<char> >&, char)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::get(char&)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::get()(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::peek()(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::sync()(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::seekg(fpos<__mbstate_t>)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::tellg()(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::unget()(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::ignore()(GLIBCXX_3.4.5) [ISOCXX]
basic_istream<char, char_traits<char> >&::sentry::sentry(basic_istream<char, char_traits<char> >&&, bool)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::putback(char)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >&::basic_istream(basic_streambuf<char, char_traits<char> >*)(GLIBCXX_3.4) [ISOCXX]
```
<table>
<thead>
<tr>
<th>Function Description</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::basic_istream()</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::basic_istream(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;*)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;)(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(ios_base&amp; (*)(ios_base&amp;))</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;*)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(void*&amp;)(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(bool&amp;)(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(double&amp;)(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(long double&amp;)(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(short&amp;)(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(unsigned short&amp;)(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(long long&amp;)(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(unsigned long long&amp;)(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; ws&lt;char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;)</td>
<td>GLIBCXX_3.4</td>
</tr>
</tbody>
</table>

**Sources:**

1. basic_istream - Generic 5.0
2. ISOCXX

**Note:** This is a list of functions and operators for the basic_istream class in the GLIBCXX_3.4 library. They are part of the ISO C++ standard library.
An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_istream<char, std::char_traits<char> > specified in Table 16-191, with the full mandatory functionality as described in the referenced underlying specification.
16.1.71 Class `basic_istream<wchar_t, char_traits<wchar_t> >`

16.1.71.1 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 in the relevant architecture specific part of this specification.

The VTT for the `std::basic_istream<wchar_t, std::char_traits<wchar_t> >` class is described by Table 16-192

### Table 16-192 VTT for `basic_istream<wchar_t, char_traits<wchar_t> >`

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_ZTTStd13basic_istreamIwSt11char_traitsIwEE</code></td>
<td></td>
</tr>
<tr>
<td>Number of Entries</td>
<td>2</td>
</tr>
</tbody>
</table>

16.1.71.2 Interfaces for Class `basic_istream<wchar_t, char_traits<wchar_t> >`

An LSB conforming implementation shall provide the generic methods for Class `std::basic_istream<wchar_t, std::char_traits<wchar_t> >` specified in Table 16-193, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-193 `libstdcxx - Class basic_istream<wchar_t, char_traits<wchar_t> >` Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>::gcount() const(GLIBCXX_3.4)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>::sentry::operator bool()</code> const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>::get(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;)(GLIBCXX_3.4)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>::get(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, wchar_t)(GLIBCXX_3.4)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>::get(wchar_t&amp;)(GLIBCXX_3.4)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>::peek() GLIBCXX_3.4)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>::sync() GLIBCXX_3.4)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>::seekg(fpos&lt;__mbstate_t&gt;) GLIBCXX_3.4)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>::tellg() GLIBCXX_3.4)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>::unget(GLIBCXX_3.4)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>::ignore(GLIBCXX_3.4)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::sentry::sentry(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, bool)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::sentry::sentry(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, bool)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::sentry::sentry(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, bool)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::putback(wchar_t)(GLIBCXX_3.4)</td>
<td>ISOCXX</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::basic_istream(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;*)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::basic_istream(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;*)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::~basic_istream()(GLIBCXX_3.4)</td>
<td>ISOCXX</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::basic_istream(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;*)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(ios_base&amp; (*)(ios_base&amp;))</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; (*)(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;))</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; (*)(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;))</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;*)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::basic_istream(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;*)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(void*&amp;)(GLIBCXX_3.4)</td>
<td>ISOCXX</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(float&amp;)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(long double&amp;)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(long&amp;)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(unsigned long&amp;)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(int&amp;)(GLIBCXX_3.4)</td>
<td>ISOCXX</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(unsigned int&amp;)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(long&amp;)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(long&amp;)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(long&amp;)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;::operator&gt;&gt;(long&amp;)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(short&amp;)</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(unsigned short&amp;)</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(long long&amp;)</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(unsigned long long&amp;)</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; ws&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; getline&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; getline&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;, wchar_t&gt;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;double, wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t, char_traits&lt;wchar_t&gt;, wchar_t&gt; &gt; &amp;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;long double, wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t, char_traits&lt;wchar_t&gt;, wchar_t&gt; &gt; &amp;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;float, wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t, char_traits&lt;wchar_t&gt;, wchar_t&gt; &gt; &amp;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, wchar_t*, _Setiosflags&gt;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, wchar_t*, _Setprecision&gt;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, wchar_t*, _Resetiosflags&gt;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, wchar_t*, _Setw&gt;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, wchar_t*, _Setbase&gt;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, wchar_t*, _Setfill&lt;wchar_t&gt;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, wchar_t*, _Setfill&lt;wchar_t&gt;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, wchar_t*, _Setfill&lt;wchar_t&gt;</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class
std::basic_istream<wchar_t, std::char_traits<wchar_t> > specified in Table 16-194,
with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>::operator++()</td>
<td>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator++() (GLIBCXX_3.4.5) [ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.72 Class istreambuf_iterator<wchar_t, char_traits<wchar_t> >

16.1.72.1 Interfaces for Class istreambuf_iterator<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the generic methods for Class std::is-
streambuf_iterator<wchar_t, std::char_traits<wchar_t> > specified in Table 16-195, with
the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>::operator++()</td>
<td>istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;::operator++() (GLIBCXX_3.4.5) [ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.73 Class istreambuf_iterator<char, char_traits<char> >

16.1.73.1 Interfaces for Class istreambuf_iterator<char, char_traits<char> >

An LSB conforming implementation shall provide the generic methods for Class std::is-
streambuf_iterator<char, std::char_traits<char> > specified in Table 16-196, with the full
mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>::operator++()</td>
<td>istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;::operator++() (GLIBCXX_3.4.5) [ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.74 Class basic_ostream<char, char_traits<char> >

16.1.74.1 Class data for basic_ostream<char, char_traits<char> >

The virtual table for the std::basic_ostream<char, std::char_traits<char> > class is de-
scribed in the relevant architecture specific part of this specification.
The VTT for the std::basic_ostream<char, std::char_traits<char>> > class is described by Table 16-197.

Table 16-197 VTT for basic_ostream<char, char_traits<char>> >

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>_ZTTS0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Entries</td>
<td>2</td>
</tr>
</tbody>
</table>

16.1.74.2 Interfaces for Class basic_ostream<char, char_traits<char>> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_ostream<char, std::char_traits<char>> > specified in Table 16-198, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-198 libstdcxx - Class basic_ostream<char, char_traits<char>> > Function Interfaces

```
<table>
<thead>
<tr>
<th>Function</th>
<th>ISOCXX</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::sentry::operator bool()</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::put(char)(GLIBCXX_3.4)</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::flush()(GLIBCXX_3.4)</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::seekp(fpos&lt;__mbstate_t&gt;)</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::tellp()(GLIBCXX_3.4)</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::sentry::sentry(basic_ostream&lt;char, char_traits&lt;char&gt;&gt;&amp;)(GLIBCXX_3.4)</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::sentry::~sentry()(GLIBCXX_3.4)</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::basic_ostream(basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;*)(GLIBCXX_3.4)</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::basic_ostream()</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::basic_ostream(basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;*)(GLIBCXX_3.4)</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::~basic_ostream()</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::~basic_ostream()</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::~basic_ostream()</td>
<td></td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::operator&lt;&lt;(basic_ostream&lt;char, char_traits&lt;char&gt;&gt;&amp; (*)(basic_ostream&lt;char, char_traits&lt;char&gt;&gt;&amp;))</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::operator&lt;&lt;(ios_base&amp; (*)(ios_base&amp;))</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &gt;::operator&lt;&lt;(basic_ios&lt;char, char_traits&lt;char&gt;&gt;&amp; (*)(basic_ios&lt;char, char_traits&lt;char&gt;&gt;&amp;))</td>
<td>GLIBCXX_3.4</td>
</tr>
</tbody>
</table>
```
<table>
<thead>
<tr>
<th>Function</th>
<th>Declaration</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (void const*)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;*)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (bool)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (double)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (long double)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (float)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (int)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (unsigned int)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (long)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (unsigned long)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (short)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (unsigned short)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (long long)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (unsigned long long)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (signed char const*)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (char const*)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (unsigned char const*)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (signed char)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (char)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (unsigned char)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (signed char const*)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (char const*)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (signed char)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (char)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (signed char const*)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (char const*)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (signed char)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (char)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt; (signed char const*)</td>
<td>(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class
std::basic_ostream<char, std::char_traits<char>> > specified in Table 16-199, with
the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Template</td>
<td>std::basic_ostream&lt;char, std::char_traits&lt;char&gt;&gt;</td>
</tr>
<tr>
<td>Function</td>
<td>&amp; operator&lt;&lt; &lt;char, char_traits&lt;char&gt; &gt;</td>
</tr>
<tr>
<td>Function</td>
<td>(basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &amp;, _Setiosflags)(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Function</td>
<td>(basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &amp;, _Setprecision)(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Function</td>
<td>(basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &amp;, _Resetiosflags)(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Function</td>
<td>(basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &amp;, _Setw)(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Function</td>
<td>(basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &amp;, _Setbase)(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Function</td>
<td>(basic_ostream&lt;char, char_traits&lt;char&gt;&gt; &amp;, _Setfill&lt;char&gt;)(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Function</td>
<td>(basic_ostream&lt;char, char_traits&lt;char&gt;&gt;, allocator&lt;char&gt;&gt; &gt; const&amp;)(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Function</td>
<td>(basic_ostream&lt;char, char_traits&lt;char&gt;&gt;, complex&lt;double&gt; const&amp;)(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Function</td>
<td>(basic_ostream&lt;char, char_traits&lt;char&gt;&gt;, complex&lt;long double&gt; const&amp;)(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Function</td>
<td>(basic_ostream&lt;char, char_traits&lt;char&gt;&gt;, complex&lt;float&gt; const&amp;)(GLIBCXX_3.4)</td>
</tr>
</tbody>
</table>

16.1.75 Class basic_ostream<wchar_t, char_traits<wchar_t>> >

16.1.75.1 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 in the relevant architecture specific part of this specification.

The VTT for the std::basic_ostream<wchar_t, std::char_traits<wchar_t>> > class is de-
Table 16-200 VTT for basic_ostream<wchar_t, char_traits<wchar_t> >

| VTT Name | 
|-----------------|-----------------|
| _ZTTSt13basic_ostreamlwSt11char_traitslwEE | 

Number of Entries 2

16.1.75.2 Interfaces for Class basic_ostream<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_ostream<wchar_t, std::char_traits<wchar_t> > specified in Table 16-201, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-201 libstdcxx - Class basic_ostream<wchar_t, char_traits<wchar_t> > Function Interfaces

| Function | 
|-----------------|-----------------|
| basic_ostream<wchar_t, char_traits<wchar_t> >::sentry::operator bool() | const(GLIBCXX_3.4) [ISOCXX] |
| basic_ostream<wchar_t, char_traits<wchar_t> >::put(wchar_t)(GLIBCXX_3.4) | [ISOCXX] |
| basic_ostream<wchar_t, char_traits<wchar_t> >::flush()(GLIBCXX_3.4) | [ISOCXX] |
| basic_ostream<wchar_t, char_traits<wchar_t> >::seekp(fpos<__mbstate_t>) | (GLIBCXX_3.4) [ISOCXX] |
| basic_ostream<wchar_t, char_traits<wchar_t> >::tellp()(GLIBCXX_3.4) | [ISOCXX] |
| basic_ostream<wchar_t, char_traits<wchar_t> >::sentry::sentry(basic_ostream<wchar_t, char_traits<wchar_t> >&) | (GLIBCXX_3.4) [ISOCXX] |
| basic_ostream<wchar_t, char_traits<wchar_t> >::sentry::sentry(basic_ostream<wchar_t, char_traits<wchar_t> >&) | (GLIBCXX_3.4) [ISOCXX] |
| basic_ostream<wchar_t, char_traits<wchar_t> >::sentry::~sentry()(GLIBCXX_3.4) | [ISOCXX] |
| basic_ostream<wchar_t, char_traits<wchar_t> >::basic_ostream(basic_streambuf<wchar_t, char_traits<wchar_t> >*) | (GLIBCXX_3.4) [ISOCXX] |
| basic_ostream<wchar_t, char_traits<wchar_t> >::basic_ostream(basic_streambuf<wchar_t, char_traits<wchar_t> >*) | (GLIBCXX_3.4) [ISOCXX] |
| basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(basic_ostream<wchar_t, char_traits<wchar_t> >& (*)(basic_ostream<wchar_t, char_traits<wchar_t> >&))(GLIBCXX_3.4) | [ISOCXX] |
| basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(ios_base& (*)(ios_base&))(GLIBCXX_3.4) | [ISOCXX] |
16 Libraries

LSB Core - Generic 5.0

```cpp
basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(basic_ios<wchar_t, char_traits<wchar_t> >& (*)(basic_ios<wchar_t, char_traits<wchar_t> >&)) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(void const*) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(basic_streambuf<wchar_t, char_traits<wchar_t> >*) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(bool) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(double) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(long double) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(float) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(int) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(unsigned int) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(long) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(unsigned long) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(short) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(unsigned short) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(long long) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >::operator<<(unsigned long long) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >& endl<wchar_t, char_traits<wchar_t> >(basic_ostream<wchar_t, char_traits<wchar_t> >&) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >& ends<wchar_t, char_traits<wchar_t> >(basic_ostream<wchar_t, char_traits<wchar_t> >&) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >& flush<wchar_t, char_traits<wchar_t> >(basic_ostream<wchar_t, char_traits<wchar_t> >&) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >& operator<< <double, wchar_t, char_traits<wchar_t> >(basic_ostream<wchar_t, char_traits<wchar_t> >&, complex<double> const&) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >& operator<< <long double, wchar_t, char_traits<wchar_t> >(basic_ostream<wchar_t, char_traits<wchar_t> >&, complex<long double> const&) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >& operator<< <float, wchar_t, char_traits<wchar_t> >(basic_ostream<wchar_t, char_traits<wchar_t> >&, complex<float> const&) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >& operator<< <wchar_t, char_traits<wchar_t> >(basic_ostream<wchar_t, char_traits<wchar_t> >&, const wchar_t*) (GLIBCXX_3.4) [ISOCXX]

basic_ostream<wchar_t, char_traits<wchar_t> >& operator<< <wchar_t, char_traits<wchar_t> >(basic_ostream<wchar_t, char_traits<wchar_t> >&, const wchar_t) (GLIBCXX_3.4) [ISOCXX]
```

664 © 2015 Linux Foundation
An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_ostream<wchar_t, std::char_traits<wchar_t> > specified in Table 16-202, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-202 libstdcxx - Class basic_ostream<wchar_t, char_traits<wchar_t> >

<table>
<thead>
<tr>
<th>Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; (GLIBCXX_3.4)</td>
</tr>
<tr>
<td>typeinfo name for basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; (GLIBCXX_3.4)</td>
</tr>
<tr>
<td>VTT for basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; (GLIBCXX_3.4)</td>
</tr>
<tr>
<td>vtable for basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; (GLIBCXX_3.4)</td>
</tr>
</tbody>
</table>
16.1.76 Class basic_fstream<char, char_traits<char> >

16.1.76.1 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 in the relevant architecture specific part of this specification.

The VTT for the std::basic_fstream<char, std::char_traits<char> > class is described by Table 16-203

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>_ZTTSt13basic_fstreamIcSt11char_traitsIcEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Entries</td>
<td>10</td>
</tr>
</tbody>
</table>

16.1.76.2 Interfaces for Class basic_fstream<char, char_traits<char> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_fstream<char, std::char_traits<char> > specified in Table 16-204, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::rdbuf() const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::is_open() const(GLIBCXX_3.4.5) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::open(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::close()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::is_open()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::basic_fstream(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::basic_fstream(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::basic_fstream&lt;char const*, _Ios_Openmode&gt;(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::basic_fstream(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::basic_fstream(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::~basic_fstream(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::~basic_fstream(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;char, char_traits&lt;char&gt; &gt;::~basic_fstream(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_fstream<char, std::char_traits<char> > specified in Table 16-205, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for basic_fstream&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI-</td>
</tr>
</tbody>
</table>
16.1.77 Class basic_fstream<wchar_t, char_traits<wchar_t> >

16.1.77.1 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 in the relevant architecture specific part of this specification.

The VTT for the std::basic_fstream<wchar_t, std::char_traits<wchar_t> > class is described by Table 16-206.

Table 16-206 VTT for basic_fstream<wchar_t, char_traits<wchar_t> >

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>_ZTTSt13basic_fstreamIwSt11char_traitsIwEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Entries</td>
<td>10</td>
</tr>
</tbody>
</table>

16.1.77.2 Interfaces for Class basic_fstream<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_fstream<wchar_t, std::char_traits<wchar_t> > specified in Table 16-207, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-207 libstdcxx - Class basic_fstream<wchar_t, char_traits<wchar_t> >

Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::rdbuf() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::is_open() const(GLIBCXX_3.4,5)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::open(char const*, _Ios_Openmode)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::close(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::is_open(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_fstream(char const*, _Ios_Openmode)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_fstream(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_fstream(char const*, _Ios_Openmode)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_fstream(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_fstream(char const*, _Ios_Openmode)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_fstream(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_fstream(char const*, _Ios_Openmode)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_fstream(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class
std::basic_fstream<wchar_t, std::char_traits<wchar_t> > specified in Table 16-208, with
the full mandatory functionality as described in the referenced underlying specification.

### Table 16-208 libstdcxx - Class basic_fstream<wchar_t, char_traits<wchar_t> > >

<table>
<thead>
<tr>
<th>Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; (GLIBCXX_3.4)</td>
</tr>
<tr>
<td>typeinfo name for basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; (GLIBCXX_3.4)</td>
</tr>
<tr>
<td>VTT for basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; (GLIBCXX_3.4)</td>
</tr>
<tr>
<td>vtable for basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; (GLIBCXX_3.4)</td>
</tr>
</tbody>
</table>

### 16.1.78 Class basic_ifstream<char, char_traits<char> >

#### 16.1.78.1 Class data for basic_ifstream<char, char_traits<char> >

The virtual table for the std::basic_ifstream<char, std::char_traits<char> > class is de-
scribed in the relevant architecture specific part of this specification.

The VTT for the std::basic_ifstream<char, std::char_traits<char> > class is described by
Table 16-209

### Table 16-209 VTT for basic_ifstream<char, char_traits<char> >

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>_ZTTSt14basic_ifstreamIcSt11char_traitsIcEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Entries</td>
<td>4</td>
</tr>
</tbody>
</table>

#### 16.1.78.2 Interfaces for Class basic_ifstream<char, char_traits<char> >

An LSB conforming implementation shall provide the generic methods for Class
std::basic_ifstream<char, std::char_traits<char> > specified in Table 16-210, with
the full mandatory functionality as described in the referenced underlying specification.

### Table 16-210 libstdcxx - Class basic_ifstream<char, char_traits<char> > > Function Interfaces

<table>
<thead>
<tr>
<th>Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::rdbuf() const(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::is_open() const(GLIBCXX_3.4.5)</td>
</tr>
<tr>
<td>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::open(char const*, _Ios_Openmode) (GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::close() (GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::is_open() (GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ifstream(char const*, _Ios_Openmode)(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ifstream() (GLIBCXX_3.4)</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_ifstream<char, std::char_traits<char> >` specified in Table 16-211, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-211 libstdcxx - Class `basic_ifstream<char, char_traits<char> >` > Data Interfaces

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>_ZTTSt14basic_ifstreamIwSt11char_traitsIwEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Entries</td>
<td>4</td>
</tr>
</tbody>
</table>

### 16.1.79 Class `basic_ifstream<wchar_t, char_traits<wchar_t> >`>

#### 16.1.79.1 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 in the relevant architecture specific part of this specification.

The VTT for the `std::basic_ifstream<wchar_t, std::char_traits<wchar_t> >` class is described by Table 16-212

### Table 16-212 VTT for `basic_ifstream<wchar_t, char_traits<wchar_t> >`>

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>_ZTTSt14basic_ifstreamIwSt11char_traitsIwEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Entries</td>
<td>4</td>
</tr>
</tbody>
</table>

#### 16.1.79.2 Interfaces for Class `basic_ifstream<wchar_t, char_traits<wchar_t> >`>

An LSB conforming implementation shall provide the generic methods for Class `std::basic_ifstream<wchar_t, std::char_traits<wchar_t> >` specified in Table 16-213, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-213 libstdcxx - Class `basic_ifstream<wchar_t, char_traits<wchar_t> >` > Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::rdbuf()</code> const(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td><code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::is_open()</code> const(GLIBCXX_3.4.5)</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_ifstream<wchar_t, std::char_traits<wchar_t> > specified in Table 16-214, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::open(char const*, _Ios_Openmode)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::close()</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::is_open()</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_ifstream(char const*, _Ios_Openmode)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_ifstream()</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_ifstream(char const*, _Ios_Openmode)</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_ifstream()</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~basic_ifstream()</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~basic_ifstream()</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~basic_ifstream()</td>
<td>GLIBCXX_3.4</td>
</tr>
</tbody>
</table>

### 16.1.80 Class basic_ofstream<char, char_traits<char> >

#### 16.1.80.1 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 in the relevant architecture specific part of this specification.

The VTT for the std::basic_ofstream<char, std::char_traits<char> > class is described by
### Table 16-215

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>_ZTTSt14basic_ofstreamIcSt11char_traitsIcEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Entries</td>
<td>4</td>
</tr>
</tbody>
</table>

### 16.1.80.2 Interfaces for Class basic_ofstream<char, char_traits<char> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_ofstream<char, std::char_traits<char> > specified in Table 16-216, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-216 libstdcxx - Class basic_ofstream<char, char_traits<char> > > Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::rdbuf() const</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::is_open() const</td>
<td>(GLIBCXX_3.4.5)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::open(char const*, _Ios_Openmode)</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::close()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::is_open()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ofstream(char const*, _Ios_Openmode)(GLIBCXX_3.4)</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ofstream()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ofstream(char const*, _Ios_Openmode)(GLIBCXX_3.4)</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ofstream()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ofstream(char const*, _Ios_Openmode)(GLIBCXX_3.4)</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ofstream()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::~basic_ofstream()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::~basic_ofstream()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::~basic_ofstream()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
</tbody>
</table>

### Table 16-217 libstdcxx - Class basic_ofstream<char, char_traits<char> > > Data Interfaces

<table>
<thead>
<tr>
<th>Data Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>typeinfo name for basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>VTT for basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>vtable for basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;</td>
<td>(GLIBCXX_3.4)</td>
</tr>
</tbody>
</table>
16.1.81 Class basic_ofstream<wchar_t, char_traits<wchar_t> >

16.1.81.1 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 in the relevant architecture specific part of this specification.

The VTT for the std::basic_ofstream<wchar_t, std::char_traits<wchar_t> > class is described by Table 16-218.

<table>
<thead>
<tr>
<th>VTT Name</th>
<th>Number of Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ZTTS14basic_ofstreamIwSt11char_traitsIwEE</td>
<td>4</td>
</tr>
</tbody>
</table>

16.1.81.2 Interfaces for Class basic_ofstream<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_ofstream<wchar_t, std::char_traits<wchar_t> > specified in Table 16-219, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::rdbuf() const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::is_open() const(GLIBCXX_3.4,5) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::open(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::close()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::is_open()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_ofstream(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_ofstream()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~basic_ofstream() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~basic_ofstream() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~basic_ofstream() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~basic_ofstream() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_ofstream<wchar_t, std::char_traits<wchar_t> > specified in Table 16-220.
with the full mandatory functionality as described in the referenced underlying specification.

Table 16-220  libstdcxx - Class basic_ofstream<wchar_t, char_traits<wchar_t> > Data Interfaces

<table>
<thead>
<tr>
<th>Data Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo</td>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;</td>
</tr>
<tr>
<td>[CXXABI-1.86]</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>typeinfo name</td>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;</td>
</tr>
<tr>
<td>[CXXABI-1.86]</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>VTT</td>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;</td>
</tr>
<tr>
<td>[CXXABI-1.86]</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>vtable</td>
<td>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;</td>
</tr>
<tr>
<td>[CXXABI-1.86]</td>
<td>GLIBCXX_3.4</td>
</tr>
</tbody>
</table>

### 16.1.82 Class basic_streambuf<char, char_traits<char> >

#### 16.1.82.1 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 16-221.

Table 16-221 Primary vtable for basic_streambuf<char, char_traits<char> >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;</td>
</tr>
<tr>
<td>vfunc[0]: basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::~basic_streambuf()</td>
<td></td>
</tr>
<tr>
<td>vfunc[1]: basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::~basic_streambuf()</td>
<td></td>
</tr>
<tr>
<td>vfunc[2]: basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::imbue(locale const&amp;)</td>
<td></td>
</tr>
<tr>
<td>vfunc[3]: See architecture specific part.</td>
<td></td>
</tr>
<tr>
<td>vfunc[4]: See architecture specific part.</td>
<td></td>
</tr>
<tr>
<td>vfunc[5]: basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::seekpos(fpos&lt;__mbstate_t&gt;, _Ios_Openmode)</td>
<td></td>
</tr>
<tr>
<td>vfunc[6]: basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::sync()</td>
<td></td>
</tr>
<tr>
<td>vfunc[7]: basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::showmanyc()</td>
<td></td>
</tr>
<tr>
<td>vfunc[8]: See architecture specific part.</td>
<td></td>
</tr>
<tr>
<td>vfunc[9]: basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::underflow()</td>
<td></td>
</tr>
<tr>
<td>vfunc[10]: basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::uflow()</td>
<td></td>
</tr>
<tr>
<td>vfunc[11]: basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::pbackfail(int)</td>
<td></td>
</tr>
<tr>
<td>vfunc[12]: See architecture specific part.</td>
<td></td>
</tr>
<tr>
<td>vfunc[13]: basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;</td>
<td></td>
</tr>
</tbody>
</table>
The Run Time Type Information for the std::basic_streambuf<char, std::char_traits<char>> class is described by Table 16-222.

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxaabi1::__class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;</td>
</tr>
</tbody>
</table>

### 16.1.82.2 Interfaces for Class basic_streambuf<char, char_traits<char>>

An LSB conforming implementation shall provide the generic methods for Class std::basic_streambuf<char, std::char_traits<char>> specified in Table 16-223, with the full mandatory functionality as described in the referenced underlying specification.

#### Table 16-223 libstdcxx - Class basic_streambuf<char, char_traits<char>>

Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>::gptr()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::pptr()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::eback()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::egptr()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::epptr()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::pbase()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::getloc()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::pubseekpos</td>
<td>(GLIBCXX_3.4, _Ios_Openmode)</td>
</tr>
<tr>
<td>::setg(char*, char*, char*)</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::setp(char*, char*)</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::sync()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::gbump(int)</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::imbue(locale const&amp;)</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::sbumpc()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::snextc()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::pubsync()</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>::seekpos(fpos&lt;__mbstate_t&gt;, fpos&lt;__mbstate_t&gt;)</td>
<td>(GLIBCXX_3.4)</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_streambuf<char, std::char_traits<char> > specified in Table 16-224, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-224 libstdcxx - Class basic_streambuf<char, char_traits<char> > > Data Interfaces

<table>
<thead>
<tr>
<th>Typeinfo for basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;</th>
<th>(GLIBCXX_3.4) [CXXABI-1.86]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typeinfo name for basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;</td>
<td>(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>Vtable for basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;</td>
<td>(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.83 Class basic_streambuf<wchar_t, char_traits<wchar_t> >

16.1.83.1 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 16-225.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::~basic_streambuf()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::~basic_streambuf()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::imbue(locale const&amp;)</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::seekpos(fpos&lt;__mbstate_t&gt;, _Ios_Openmode)</td>
</tr>
<tr>
<td>vfunc[6]:</td>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::sync()</td>
</tr>
<tr>
<td>vfunc[7]:</td>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::showmanyc()</td>
</tr>
<tr>
<td>vfunc[8]:</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>vfunc[9]:</td>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::underflow()</td>
</tr>
<tr>
<td>vfunc[10]:</td>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::uflow()</td>
</tr>
<tr>
<td>vfunc[11]:</td>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::pbackfail(unsigned int)</td>
</tr>
<tr>
<td>vfunc[12]:</td>
<td>See architecture specific part.</td>
</tr>
<tr>
<td>vfunc[13]:</td>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::overflow(unsigned int)</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::basic_streambuf<wchar_t, char_traits<wchar_t>> class is described by Table 16-226.

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</td>
</tr>
</tbody>
</table>
16.1.83.2 Interfaces for Class basic_streambuf<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_streambuf<wchar_t, std::char_traits<wchar_t> > specified in Table 16-227, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-227 libstdcxx - Class basic_streambuf<wchar_t, char_traits<wchar_t> >

<table>
<thead>
<tr>
<th>Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::gptr() const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::pptr() const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::eback() const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::egptr() const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::epptr() const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::pbase() const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::getloc() const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::pubseekpos(fpos&lt;__mbstate_t&gt;, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::setg(wchar_t*, wchar_t*, wchar_t*)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::setp(wchar_t*, wchar_t*) (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::sync()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::gbump(int)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::imbue(locale const&amp;) (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::pbump(int)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::sgetc() (GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::sputc(wchar_t)(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::uflow()(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::sbumpc()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::snextc()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::pubsync()(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::seekpos(fpos&lt;__mbstate_t&gt;,</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
---

### 16.1.84 Class `basic_filebuf<char, char_traits<char>>` >

#### 16.1.84.1 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 16-228.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td><code>typeid for </code>basic_filebuf&lt;char, char_traits&lt;char&gt;&gt;` &gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td><code>basic_filebuf&lt;char, char_traits&lt;char&gt;&gt;</code> &gt; ::~basic_filebuf()</td>
</tr>
</tbody>
</table>

---
** LSB Core - Generic 5.0  

| vfunc[1]: basic_filebuf<char, char_traits<char> >::~basic_filebuf() |
| vfunc[2]: basic_filebuf<char, char_traits<char> >::imbue(locale const&) |
| vfunc[3]: See architecture specific part. |
| vfunc[4]: See architecture specific part. |
| 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]: See architecture specific part. |
| 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]: See architecture specific part. |
| 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 16-229**

**Table 16-229 typeinfo for basic_filebuf<char, char_traits<char> >**

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;</td>
</tr>
</tbody>
</table>

**16.1.84.2 Interfaces for Class basic_filebuf<char, char_traits<char> >**

An LSB conforming implementation shall provide the generic methods for Class std::basic_filebuf<char, std::char_traits<char> > specified in **Table 16-230**, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-230 libstdcxx - Class basic_filebuf<char, char_traits<char> > > Function Interfaces**

| basic_filebuf<char, char_traits<char> > >::is_open() const(GLIBCXX_3.4) | [ISOCXX] |
| basic_filebuf<char, char_traits<char> > >::_M_create_pback() | (GLIBCXX_3.4) | [ISOCXX] |
| basic_filebuf<char, char_traits<char> > >::_M_destroy_pback() | (GLIBCXX_3.4) | [ISOCXX] |
| basic_filebuf<char, char_traits<char> > >::_M_terminate_output() | (GLIBCXX_3.4) | [ISOCXX] |
| basic_filebuf<char, char_traits<char> > >::_M_destroy_internal_buffer() | (GLIBCXX_3.4) | [ISOCXX] |
| basic_filebuf<char, char_traits<char> > >::_M_allocate_internal_buffer() | (GLIBCXX_3.4) | [ISOCXX] |
An LSB conforming implementation shall provide the generic data interfaces for Class
std::basic_filebuf<char, std::char_traits<char> > specified in Table 16-231, with the full
mandatory functionality as described in the referenced underlying specification.

Table 16-231  libstdcxx - Class basic_filebuf<char, char_traits<char> > > Data Interfaces

typeinfo for basic_filebuf<char, char_traits<char> > >(GLIBCXX_3.4) [CXXABI-1.86]
typeinfo name for basic_filebuf<char, char_traits<char> > >(GLIBCXX_3.4) [CXXABI-1.86]
vtable for basic_filebuf<char, char_traits<char> > >(GLIBCXX_3.4) [CXXABI-1.86]

16.1.85 Class basic_filebuf<wchar_t, char_traits<wchar_t> >

16.1.85.1 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 16-232.

Table 16-232 Primary vtable for basic_filebuf<wchar_t, char_traits<wchar_t> >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; ::~basic_filebuf()</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the std::basic_filebuf<wchar_t, std::char_traits<wchar_t> > class is described by Table 16-233.

### Table 16-233 typeinfo for basic_filebuf<wchar_t, char_traits<wchar_t> >

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt;</td>
</tr>
</tbody>
</table>

### 16.1.85.2 Interfaces for Class basic_filebuf<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_filebuf<wchar_t, std::char_traits<wchar_t> > specified in Table 16-234, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-234 libstdcxx - Class basic_filebuf<wchar_t, char_traits<wchar_t> > Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::is_open() const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::M_create_pback()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::M_destroy_pback(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

---

© 2015 Linux Foundation
An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_filebuf<wchar_t, std::char_traits<wchar_t>> as specified in Table 16-235, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-235 libstdcxx - Class basic_filebuf<wchar_t, char_traits<wchar_t>> > Data Interfaces

typeinfo for basic_filebuf<wchar_t, char_traits<wchar_t>> > (GLIBCXX_3.4) [CXXABI-1.86]
typeinfo name for basic_filebuf<wchar_t, char_traits<wchar_t>> > (GLIBCXX_3.4)
16.1.86 Class \texttt{ios\_base}

16.1.86.1 Class data for \texttt{ios\_base}

The virtual table for the \texttt{std::ios\_base} class is described by Table 16-236.

**Table 16-236 Primary vtable for \texttt{ios\_base}**

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>\texttt{typeinfo for ios_base}</td>
</tr>
<tr>
<td>\texttt{vfunc[0]}</td>
<td>\texttt{ios_base::~ios_base()}</td>
</tr>
<tr>
<td>\texttt{vfunc[1]}</td>
<td>\texttt{ios_base::~ios_base()}</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the \texttt{std::ios\_base} class is described by Table 16-237.

**Table 16-237 typeinfo for \texttt{ios\_base}**

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1__class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for \texttt{ios_base}</td>
</tr>
</tbody>
</table>

16.1.86.2 Interfaces for Class \texttt{ios\_base}

An LSB conforming implementation shall provide the generic methods for Class \texttt{std::ios\_base} specified in Table 16-238, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-238 libstdcxx - Class \texttt{ios\_base} Function Interfaces**

| \texttt{ios\_base::\_M\_grow\_words(int, bool)}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base::sync\_with\_stdio(bool)}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base:\_M\_call\_callbacks(ios\_base::event)(GLIBCXX 3.4.6)} | [ISOCXX] |
| \texttt{ios\_base::register\_callback(void (*)(ios\_base::event, ios\_base&, int, int)}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base:\_M\_dispose\_callbacks()}(GLIBCXX 3.4.6) | [ISOCXX] |
| \texttt{ios\_base::Init::Init()}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base::Init::Init()}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base::Init::~Init()}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base::Init::~Init()}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base::imbue(locale const&)}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base::xalloc()}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base::\_M\_init()}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base::failure::failure(basic\_string\<char, char\_traits\<char>, allocator\<char\> > const&)}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base::failure::failure(basic\_string\<char, char\_traits\<char>, allocator\<char\> > const&)}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base::failure::~failure()}(GLIBCXX 3.4) | [ISOCXX] |
| \texttt{ios\_base::failure::~failure()}(GLIBCXX 3.4) | [ISOCXX] |
An LSB conforming implementation shall provide the generic data interfaces for Class `std::ios_base` specified in Table 16-239, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-239 libstdcxx - Class `ios_base` Data Interfaces**

<table>
<thead>
<tr>
<th>Function</th>
<th>Libcxx API</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>floatfield</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>scientific</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>adjustfield</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>in</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>app</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>ate</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>beg</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>cur</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>dec</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>end</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>hex</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>oct</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>out</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>left</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>fixed</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>right</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>trunc</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>badbit</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>binary</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>eofbit</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>skipws</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>failbit</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>goodbit</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>showpos</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>unitbuf</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>internal</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>showbase</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>basefield</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>boolalpha</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>showpoint</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>uppercase</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>typeinfo for ios_base</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for ios_base</code></td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>
16.1.87 Class basic_ios<char, char_traits<char> >

16.1.87.1 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 by Table 16-240.

Table 16-240 Primary vtable for basic_ios<char, char_traits<char> >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for basic_ios&lt;char, char_traits&lt;char&gt; &gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::~basic_ios()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::~basic_ios()</td>
</tr>
</tbody>
</table>

16.1.87.2 Interfaces for Class basic_ios<char, char_traits<char> >

An LSB conforming implementation shall provide the generic methods for Class std::basic_ios<char, std::char_traits<char> > specified in Table 16-241, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-241 libstdcxx - Class basic_ios<char, char_traits<char> > Function Interfaces

| basic_ios<char, char_traits<char> >::exceptions() const(GLIBCXX_3.4) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::bad() const(GLIBCXX_3.4) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::eof() const(GLIBCXX_3.4) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::tie() const(GLIBCXX_3.4) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::fail() const(GLIBCXX_3.4) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::fill() const(GLIBCXX_3.4) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::good() const(GLIBCXX_3.4) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::operator void*() const(GLIBCXX_3.4) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::operator!() const(GLIBCXX_3.4) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::exceptions(_Ios_Iostate) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::_M_setstate(_Ios_Iostate) | [GLIBCXX_3.4] |
| basic_ios<char, char_traits<char> >::tie(basic_ostream<char, char_traits<char> >*) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::fill(char) | [ISOCXX] |
| basic_ios<char, char_traits<char> >::init(basic_streambuf<char, char_traits<char> >*) | [GLIBCXX_3.4] |
| basic_ios<char, char_traits<char> >::rdstate() | [ISOCXX] |
An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_ios<char, std::char_traits<char> > specified in Table 16-242, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-242 lbdstdc++ - Class basic_ios<char, char_traits<char> > > Data Interfaces

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for basic_ios&lt;char, char_traits&lt;char&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::~basic_ios()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::~basic_ios()</td>
</tr>
</tbody>
</table>

16.1.88 Class basic_ios<wchar_t, char_traits<wchar_t> >

16.1.88.1 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 by Table 16-243.

### Table 16-243 Primary vtable for basic_ios<wchar_t, char_traits<wchar_t> >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTTI</td>
<td>typeinfo for basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~basic_ios()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~basic_ios()</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::basic_ios<wchar_t, std::char_traits<wchar_t> > class is described by Table 16-244.
Table 16-244 typeinfo for basic_ios<wchar_t, char_traits<wchar_t>>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_t type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</td>
</tr>
<tr>
<td>flags:</td>
<td>8</td>
</tr>
<tr>
<td>basetype:</td>
<td>typeinfo for ios_base 1026</td>
</tr>
</tbody>
</table>

16.1.88.2 Interfaces for Class basic_ios<wchar_t, char_traits<wchar_t>>

An LSB conforming implementation shall provide the generic methods for Class std::basic_ios<wchar_t, std::char_traits<wchar_t>> specified in Table 16-245, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-245 libstdcxx - Class basic_ios<wchar_t, char_traits<wchar_t>>

Function Interfaces

| basic_ios<wchar_t, char_traits<wchar_t>>::exceptions() const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::bad() const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::eof() const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::tie() const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::fail() const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::fill() const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::good() const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::rdbuf() const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::widen(char) const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::narrow(wchar_t, char) const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::rdstate() const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::operator void*() const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::operator!() const(GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::exceptions(_Ios_Iostate) (GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::_M_setstate(_Ios_Iostate) (GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::tie(basic_ostream<wchar_t, char_traits<wchar_t>>*) (GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::fill(wchar_t) (GLIBCXX_3.4) [ISOCXX] |
| basic_ios<wchar_t, char_traits<wchar_t>>::init(basic_streambuf<wchar_t, char_traits<wchar_t>>*) (GLIBCXX_3.4) [ISOCXX] |
An LSB conforming implementation shall provide the generic data interfaces for Class std::basic_ios<wchar_t, std::char_traits<wchar_t> > specified in Table 16-246, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-246  libstdc++ - Class basic_ios<wchar_t, char_traits<wchar_t> > > Data Interfaces

<table>
<thead>
<tr>
<th>Typeinfo for basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;(GLIBCXX_3.4)</th>
<th>[CXXABI-1.86]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typeinfo name for basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>Vtable for basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.89  Class ios_base::failure

16.1.89.1  Class data for ios_base::failure

The virtual table for the std::ios_base::failure class is described by Table 16-247.

Table 16-247  Primary vtable for ios_base::failure

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for ios_base::failure</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>ios_base::failure::~failure()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>ios_base::failure::failure()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>ios_base::failure::what() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::ios_base::failure class is described by Table 16-248.
Table 16-248 typeinfo for ios_base::failure

| Base Vtable | vtable for
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>_cxxabi1::__si_class_type_info</td>
</tr>
</tbody>
</table>

16.1.89.2 Interfaces for Class ios_base::failure

An LSB conforming implementation shall provide the generic methods for Class std::ios_base::failure specified in Table 16-249, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-249 libstdcxx - Class ios_base::failure Function Interfaces

| ios_base::failure::what() const(GLIBCXX_3.4) [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::ios_base::failure specified in Table 16-250, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-250 libstdcxx - Class ios_base::failure Data Interfaces

| typeinfo for ios_base::failure(GLIBCXX_3.4) [CXXABI-1.86] |
| vtable for ios_base::failure(GLIBCXX_3.4) [CXXABI-1.86] |

16.1.90 Class __timepunct<char>

16.1.90.1 Class data for __timepunct<char>

The virtual table for the std::__timepunct<char> class is described by Table 16-251.

Table 16-251 Primary vtable for __timepunct<char>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>vtable for __timepunct&lt;char&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>__timepunct&lt;char&gt;::~__timepunct()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>__timepunct&lt;char&gt;::~__timepunct()</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::__timepunct<char> class is described by Table 16-252.

Table 16-252 typeinfo for __timepunct<char>

| Base Vtable | vtable for
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>_cxxabi1::__si_class_type_info</td>
</tr>
</tbody>
</table>

16.1.90.2 Interfaces for Class __timepunct<char>

An LSB conforming implementation shall provide the generic methods for Class std::__timepunct<char> specified in Table 16-253, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-253 libstdcxx - Class __timepunct<char> Function Interfaces

| __timepunct<char>::_M_am_pm_format(char const*) const(GLIBCXX_3.4) [ISOCXX] |
| __timepunct<char>::_M_date_formats(char const**) const(GLIBCXX_3.4) [ISOCXX] |
An LSB conforming implementation shall provide the generic data interfaces for Class std::__timepunct<char> specified in Table 16-254, with the full mandatory functionality as described in the referenced underlying specification.

### 16.1.91 Class __timepunct<wchar_t>
#### 16.1.91.1 Class data for __timepunct<wchar_t>

The virtual table for the std::__timepunct<wchar_t> class is described by Table 16-255.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for __timepunct&lt;wchar_t&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>__timepunct&lt;wchar_t&gt;::~__timepunct()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>__timepunct&lt;wchar_t&gt;::~__timepunct()</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::__timepunct<wchar_t> class is described by Table 16-256.

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
</table>
16.1.91.2 Interfaces for Class __timepunct<wchar_t>

An LSB conforming implementation shall provide the generic methods for Class std::__timepunct<wchar_t> specified in Table 16-257, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-257 libstdcxx - Class __timepunct<wchar_t> Function Interfaces

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>__timepunct&lt;wchar_t&gt;::_M_date_formats(wchar_t const**) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__timepunct&lt;wchar_t&gt;::_M_time_formats(wchar_t const**) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__timepunct&lt;wchar_t&gt;::_M_days_abbreviated(wchar_t const**) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__timepunct&lt;wchar_t&gt;::_M_date_time_formats(wchar_t const**) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__timepunct&lt;wchar_t&gt;::_M_months_abbreviated(wchar_t const**) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__timepunct&lt;wchar_t&gt;::_M_days(wchar_t const**) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__timepunct&lt;wchar_t&gt;::_M_am_pm(wchar_t const**) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__timepunct&lt;wchar_t&gt;::_M_months(wchar_t const**) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>__timepunct&lt;wchar_t&gt;::_M_initialize_timepunct(__locale_struct*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::__timepunct<wchar_t> specified in Table 16-258, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-258 libstdcxx - Class __timepunct<wchar_t> Data Interfaces

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guard variable for __timepunct&lt;wchar_t&gt;::id(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>__timepunct&lt;wchar_t&gt;::id(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>typeid for __timepunct&lt;wchar_t&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for __timepunct&lt;wchar_t&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for __timepunct&lt;wchar_t&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.92 Class messages_base

16.1.92.1 Class data for messages_base

The Run Time Type Information for the std::messages_base class is described by Table 16-259.
Table 16-259 typeinfo for messages_base

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for messages_base</td>
</tr>
</tbody>
</table>

16.1.92.2 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for Class std::messages_base specified in Table 16-260, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-260 libstdcxx - Class messages_base Data Interfaces

<table>
<thead>
<tr>
<th>typeinfo for messages_base(GLIBCXX_3.4) [CXXABI-1.86]</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo name for messages_base(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.93 Class messages<char>

16.1.93.1 Class data for messages<char>

The virtual table for the std::messages<char> class is described by Table 16-261.

Table 16-261 Primary vtable for messages<char>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for messages&lt;char&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>messages&lt;char&gt;::~messages()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>messages&lt;char&gt;::~messages()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>messages&lt;char&gt;::do_open(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;, locale const&amp;) const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>messages&lt;char&gt;::do_get(int, int, int, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;) const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>messages&lt;char&gt;::do_close(int) const</td>
</tr>
</tbody>
</table>

16.1.93.2 Interfaces for Class messages<char>

An LSB conforming implementation shall provide the generic methods for Class std::messages<char> specified in Table 16-262, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-262 libstdcxx - Class messages<char> Function Interfaces

<table>
<thead>
<tr>
<th>messages&lt;char&gt;::_M_convert_to_char(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;) const(GLIBCXX_3.4) [ISOCXX]</th>
</tr>
</thead>
<tbody>
<tr>
<td>messages&lt;char&gt;::_M_convert_from_char(char*) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>messages&lt;char&gt;::get(int, int, int, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>messages&lt;char&gt;::open(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;, locale const&amp;) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>messages&lt;char&gt;::open(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;, char const*) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class std::messages<char> specified in Table 16-263, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-263 libstdcxx - Class messages<char> Data Interfaces

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guard variable for messages&lt;char&gt;::id</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td>messages&lt;char&gt;::id</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo for messages&lt;char&gt;</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for messages&lt;char&gt;</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for messages&lt;char&gt;</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

### 16.1.94 Class messages<wchar_t>

#### 16.1.94.1 Class data for messages<wchar_t>

The virtual table for the std::messages<wchar_t> class is described by Table 16-264.

### Table 16-264 Primary vtable for messages<wchar_t>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>vtable for messages&lt;wchar_t&gt;</td>
<td>messages&lt;wchar_t&gt;::~messages()</td>
<td>messages&lt;wchar_t&gt;::~messages()</td>
<td>messages&lt;wchar_t&gt;::do_open(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;) const</td>
<td>messages&lt;wchar_t&gt;::do_get(int, int, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;) const</td>
<td>messages&lt;wchar_t&gt;::do_close(int) const</td>
</tr>
</tbody>
</table>

#### 16.1.94.2 Interfaces for Class messages<wchar_t>

An LSB conforming implementation shall provide the generic methods for Class std::messages<wchar_t> specified in Table 16-265, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-265 libstdcxx - Class messages<wchar_t> Function Interfaces

messages<wchar_t>::__M_convert_to_char(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&) const(GLIBCXX_3.4)
An LSB conforming implementation shall provide the generic data interfaces for Class
std::messages<wchar_t> specified in Table 16-266, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-266 libstdcxx - Class messages<wchar_t> Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>messages&lt;wchar_t&gt;::id(GLIBCXX_3.4)</td>
<td>Guard variable for messages&lt;wchar_t&gt;::id(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>messages&lt;wchar_t&gt;::typeinfo(GLIBCXX_3.4)</td>
<td>Typeinfo for messages&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>messages&lt;wchar_t&gt;::typeinfo_name(GLIBCXX_3.4)</td>
<td>Typeinfo name for messages&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for messages&lt;wchar_t&gt;(GLIBCXX_3.4)</td>
<td>Vtable for messages&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

### 16.1.95 Class messages__byname<char>

#### 16.1.95.1 Class data for messages__byname<char>

The virtual table for the std::messages__byname<char> class is described by Table 16-267.

### Table 16-267 Primary vtable for messages__byname<char>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>Typeinfo for messages__byname&lt;char&gt;</td>
</tr>
<tr>
<td>vfunc[0]</td>
<td>messages__byname&lt;char&gt;::~messages__byname()</td>
</tr>
<tr>
<td>vfunc[1]</td>
<td>messages__byname&lt;char&gt;::~messages__byname()</td>
</tr>
<tr>
<td>vfunc[2]</td>
<td>messages&lt;char&gt;::do_open(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;, locale const&amp;) const</td>
</tr>
</tbody>
</table>
| vfunc[3]     | messages<char>::do_get(int, int, int,
The Run Time Type Information for the std::messages_byname<char> class is described by Table 16-268.

Table 16-268 typeinfo for messages_byname<char>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for messages_byname&lt;char&gt;</td>
</tr>
</tbody>
</table>

16.1.95.2 Interfaces for Class messages_byname<char>

An LSB conforming implementation shall provide the generic methods for Class std::messages_byname<char> specified in Table 16-269, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-269 libstdcxx - Class messages_byname<char> Function Interfaces

| messages_byname<char>::~messages_byname()<GLIBCXX_3.4) [ISOCXX] |
| messages_byname<char>::~messages_byname()<GLIBCXX_3.4) [ISOCXX] |
| messages_byname<char>::~messages_byname()<GLIBCXX_3.4) [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::messages_byname<char> specified in Table 16-270, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-270 libstdcxx - Class messages_byname<char> Data Interfaces

| typeinfo for messages_byname<char>(GLIBCXX_3.4) [CXXABI-1.86] |
| typeinfo name for messages_byname<char>(GLIBCXX_3.4) [CXXABI-1.86] |
| vtable for messages_byname<char>(GLIBCXX_3.4) [CXXABI-1.86] |

16.1.96 Class messages_byname<wchar_t>

16.1.96.1 Class data for messages_byname<wchar_t>

The virtual table for the std::messages_byname<wchar_t> class is described by Table 16-271.

Table 16-271 Primary vtable for messages_byname<wchar_t>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for messages_byname&lt;wchar_t&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>messages_byname&lt;wchar_t&gt;::~messages_byname()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>messages_byname&lt;wchar_t&gt;::~messages_byname()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>messages&lt;wchar_t&gt;::do_open(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;) const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>messages&lt;wchar_t&gt;::do_get(int, int, int,</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the std::messages_byname<wchar_t> class is described by Table 16-272.

**Table 16-272 typeinfo for messages_byname<wchar_t>**

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabi::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for messages_byname&lt;wchar_t&gt;</td>
</tr>
</tbody>
</table>

**16.1.96.2 Interfaces for Class messages_byname<wchar_t>**

An LSB conforming implementation shall provide the generic methods for Class std::messages_byname<wchar_t> specified in Table 16-273, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-273 libstdcxx - Class messages_byname<wchar_t> Function Interfaces**

| messages_byname<wchar_t>::~messages_byname() (GLIBCXX_3.4) [ISOCXX] |
| messages_byname<wchar_t>::~messages_byname() (GLIBCXX_3.4) [ISOCXX] |
| messages_byname<wchar_t>::~messages_byname() (GLIBCXX_3.4) [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::messages_byname<wchar_t> specified in Table 16-274, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-274 libstdcxx - Class messages_byname<wchar_t> Data Interfaces**

| typeinfo name for messages_byname<wchar_t>(GLIBCXX_3.4) [CXXABI-1.86] |
| vtable for messages_byname<wchar_t>(GLIBCXX_3.4) [CXXABI-1.86] |

**16.1.97 Class numpunct<char>**

**16.1.97.1 Class data for numpunct<char>**

The virtual table for the std::numpunct<char> class is described by Table 16-275.

**Table 16-275 Primary vtable for numpunct<char>**

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for numpunct&lt;char&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>numpunct&lt;char&gt;::~numpunct()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>numpunct&lt;char&gt;::~numpunct()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>numpunct&lt;char&gt;::do_decimal_point() const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>numpunct&lt;char&gt;::do_thousands_sep() const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>numpunct&lt;char&gt;::do_grouping() const</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>numpunct&lt;char&gt;::do_truename() const</td>
</tr>
<tr>
<td>vfunc[6]:</td>
<td>numpunct&lt;char&gt;::do_falsename() const</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the std::numpunct<char> class is described by Table 16-276.

Table 16-276 typeinfo for numpunct<char>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for numpunct&lt;char&gt;</td>
</tr>
</tbody>
</table>

16.1.97.2 Interfaces for Class numpunct<char>

An LSB conforming implementation shall provide the generic methods for Class std::numpunct<char> specified in Table 16-277, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-277 libstdcxx - Class numpunct<char> Function Interfaces

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numpunct&lt;char&gt;::do_grouping() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::do_truename() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::do_falsename() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::decimal_point() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::thousands_sep() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::do_decimal_point() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::do_thousands_sep() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::grouping() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::truename() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::falsename() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::_M_initialize_numpunct(__locale_struct*)(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::~numpunct()(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::numpunct<char> specified in Table 16-278, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-278 libstdcxx - Class numpunct<char> Data Interfaces

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guard variable for numpunct&lt;char&gt;::id(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>numpunct&lt;char&gt;::id(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>typeinfo for numpunct&lt;char&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for numpunct&lt;char&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.98 Class numpunct<wchar_t>

16.1.98.1 Class data for numpunct<wchar_t>

The virtual table for the std::numpunct<wchar_t> class is described by Table 16-279.

Table 16-279 Primary vtable for numpunct<wchar_t>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
16 Libraries

LSB Core - Generic 5.0

<table>
<thead>
<tr>
<th>RTTI</th>
<th>vfunc[0]: numpunct&lt;wchar_t&gt;::~numpunct()</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>vfunc[1]: numpunct&lt;wchar_t&gt;::~numpunct()</td>
</tr>
<tr>
<td></td>
<td>vfunc[2]: numpunct&lt;wchar_t&gt;::do_decimal_point() const</td>
</tr>
<tr>
<td></td>
<td>vfunc[3]: numpunct&lt;wchar_t&gt;::do_thousands_sep() const</td>
</tr>
<tr>
<td></td>
<td>vfunc[4]: numpunct&lt;wchar_t&gt;::do_grouping() const</td>
</tr>
<tr>
<td></td>
<td>vfunc[5]: numpunct&lt;wchar_t&gt;::do_truename() const</td>
</tr>
<tr>
<td></td>
<td>vfunc[6]: numpunct&lt;wchar_t&gt;::do_falsename() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::numpunct<wchar_t> class is described by Table 16-280.

Table 16-280 typeinfo for numpunct<wchar_t>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for numpunct&lt;wchar_t&gt;</td>
</tr>
</tbody>
</table>

16.1.98.2 Interfaces for Class numpunct<wchar_t>

An LSB conforming implementation shall provide the generic methods for Class std::numpunct<wchar_t> specified in Table 16-281, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-281 libstdcxx - Class numpunct<wchar_t> Function Interfaces

numpunct<wchar_t>::do_grouping() const(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::do_truename() const(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::do_falsename() const(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::decimal_point() const(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::do_thousands_sep() const(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::do_grouping() const(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::do_truename() const(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::falsename() const(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::M_initialize_numpunct(__locale_struct*)(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::~numpunct() (GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::~numpunct() (GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::~numpunct() (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class std::numpunct<wchar_t> specified in Table 16-282, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-282 libstdcxx - Class numpunct<wchar_t> Data Interfaces

guard variable for numpunct<wchar_t>::id(GLIBCXX_3.4) [CXXABI-1.86]
16.1.99 Class numpunct_byname<char>

16.1.99.1 Class data for numpunct_byname<char>

The virtual table for the std::numpunct_byname<char> class is described by Table 16-283.

Table 16-283 Primary vtable for numpunct_byname<char>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>typeinfo for numpunct_byname&lt;char&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>numpunct_byname&lt;char&gt;::~numpunct_byname()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>numpunct_byname&lt;char&gt;::~numpunct_byname()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>numpunct&lt;char&gt;::do_decimal_point() const</td>
</tr>
<tr>
<td></td>
<td></td>
<td>numpunct&lt;char&gt;::do_Grouping() const</td>
</tr>
<tr>
<td></td>
<td></td>
<td>numpunct&lt;char&gt;::do_truename() const</td>
</tr>
<tr>
<td></td>
<td></td>
<td>numpunct&lt;char&gt;::do_falsename() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::numpunct_byname<char> class is described by Table 16-284.

Table 16-284 typeinfo for numpunct_byname<char>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for numpunct_byname&lt;char&gt;</td>
</tr>
</tbody>
</table>

16.1.99.2 Interfaces for Class numpunct_byname<char>

An LSB conforming implementation shall provide the generic methods for Class std::numpunct_byname<char> specified in Table 16-285, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-285 libstdc++ - Class numpunct_byname<char> Function Interfaces

| numpunct_byname<char>::~numpunct_byname()(GLIBCXX_3.4) | [ISO CXX] |
| numpunct_byname<char>::~numpunct_byname()(GLIBCXX_3.4) | [ISO CXX] |
| numpunct_byname<char>::~numpunct_byname()(GLIBCXX_3.4) | [ISO CXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::numpunct_byname<char> specified in Table 16-286, with the full mandatory functionality as described in the referenced underlying specification.
Table 16-286 libstdcxx - Class numpunct_byname<char> Data Interfaces

<table>
<thead>
<tr>
<th>Data Interfaces</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for numpunct_byname&lt;char&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for numpunct_byname&lt;char&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for numpunct_byname&lt;char&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.100 Class numpunct_byname<wchar_t>

16.1.100.1 Class data for numpunct_byname<wchar_t>

The virtual table for the std::numpunct_byname<wchar_t> class is described by Table 16-287.

Table 16-287 Primary vtable for numpunct_byname<wchar_t>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td>numpunct_byname&lt;wchar_t&gt;::~numpunct_byname()</td>
<td>numpunct_byname&lt;wchar_t&gt;::~numpunct_byname()</td>
<td>numpunct&lt;wchar_t&gt;::do_decimal_point() const</td>
<td>numpunct&lt;wchar_t&gt;::doThousandsSep() const</td>
<td>numpunct&lt;wchar_t&gt;::doGrouping() const</td>
<td>numpunct&lt;wchar_t&gt;::doTrueName() const</td>
<td>numpunct&lt;wchar_t&gt;::doFalseName() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::numpunct_byname<wchar_t> class is described by Table 16-288.

Table 16-288 typeinfo for numpunct_byname<wchar_t>

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vtable for __cxaabi1::__si_class_type_info</td>
<td>typeinfo name for numpunct_byname&lt;wchar_t&gt;</td>
</tr>
</tbody>
</table>

16.1.100.2 Interfaces for Class numpunct_byname<wchar_t>

An LSB conforming implementation shall provide the generic data interfaces for Class std::numpunct_byname<wchar_t> specified in Table 16-289, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-289 libstdcxx - Class numpunct_byname<wchar_t> Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numpunct_byname&lt;wchar_t&gt;::~numpunct_byname()</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct_byname&lt;wchar_t&gt;::~numpunct_byname()</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>numpunct_byname&lt;wchar_t&gt;::~numpunct_byname()</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class
std::numpunct_byname<wchar_t> specified in Table 16-290, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-290 libstdcxx - Class numpunct_byname<wchar_t> Data Interfaces

| typeinfo for numpunct_byname<wchar_t>(GLIBCXX_3.4) [CXXABI-1.86] |
| vtable for numpunct_byname<wchar_t>(GLIBCXX_3.4) [CXXABI-1.86] |

16.1.101 Class __codecvt_abstract_base<char, char, __mbstate_t>

16.1.101.1 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 by Table 16-291.

Table 16-291 Primary vtable for __codecvt_abstract_base<char, char, __mbstate_t>

| Base Offset | 0 |
| Virtual Base Offset | 0 |
| RTTI | typeinfo for __codecvt_abstract_base<char, char, __mbstate_t> |
| vfunc[0]: | NULL or __codecvt_abstract_base<char, char, __mbstate_t>::~__codecvt_abstract_base() |
| vfunc[1]: | NULL or __codecvt_abstract_base<char, char, __mbstate_t>::~__codecvt_abstract_base() |
| vfunc[2]: | __cxa_pure_virtual |
| vfunc[3]: | __cxa_pure_virtual |
| vfunc[4]: | __cxa_pure_virtual |
| vfunc[5]: | __cxa_pure_virtual |
| vfunc[6]: | __cxa_pure_virtual |
| vfunc[7]: | __cxa_pure_virtual |
| vfunc[8]: | __cxa_pure_virtual |

16.1.101.2 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for Class std::__codecvt_abstract_base<char, char, __mbstate_t> specified in Table 16-292, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-292 libstdcxx - Class __codecvt_abstract_base<char, char, __mbstate_t> Data Interfaces

| typeinfo for __codecvt_abstract_base<char, char, __mbstate_t>(GLIBCXX_3.4) |

© 2015 Linux Foundation
16.1.102 Class __codecvt_abstract_base<wchar_t, char, __mbstate_t>

16.1.102.1 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 by Table 16-293.

Table 16-293 Primary vtable for __codecvt_abstract_base<wchar_t, char, __mbstate_t>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RTTI</th>
<th>typeinfo for __codecvt_abstract_base&lt;wchar_t, char, __mbstate_t&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>vfunc[0]:</td>
<td>NULL or __codecvt_abstract_base&lt;wchar_t, char, __mbstate_t&gt;::~__codecvt_abstract_base()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>NULL or __codecvt_abstract_base&lt;wchar_t, char, __mbstate_t&gt;::~__codecvt_abstract_base()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>__cxa_pure_virtual</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>__cxa_pure_virtual</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>__cxa_pure_virtual</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>__cxa_pure_virtual</td>
</tr>
<tr>
<td>vfunc[6]:</td>
<td>__cxa_pure_virtual</td>
</tr>
<tr>
<td>vfunc[7]:</td>
<td>__cxa_pure_virtual</td>
</tr>
<tr>
<td>vfunc[8]:</td>
<td>__cxa_pure_virtual</td>
</tr>
</tbody>
</table>

16.1.102.2 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for Class std::__codecvt_abstract_base<wchar_t, char, __mbstate_t> specified in Table 16-294, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-294 libstdcxx - Class __codecvt_abstract_base<wchar_t, char, __mbstate_t> Data Interfaces

<table>
<thead>
<tr>
<th>typeinfo for __codecvt_abstract_base&lt;wchar_t, char, __mbstate_t&gt;(GLIBCXX_3.4)</th>
<th>[CXXABI-1.86]</th>
</tr>
</thead>
</table>
16.1.103 Class codecvt_base

16.1.103.1 Class data for codecvt_base

The Run Time Type Information for the std::codecvt_base class is described by Table 16-295.

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for codecvt_base</td>
</tr>
</tbody>
</table>

16.1.103.2 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for Class std::codecvt_base specified in Table 16-296, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>typeinfo name for codecvt_base(GLIBCXX_3.4) [CXXABI-1.86]</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo name for codecvt_base(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.104 Class codecvt<char, char, __mbstate_t>

16.1.104.1 Class data for codecvt<char, char, __mbstate_t>

The virtual table for the std::codecvt<char, char, __mbstate_t> class is described by Table 16-297.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for codecvt&lt;char, char, __mbstate_t&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>codecvt&lt;char, char, __mbstate_t&gt;::~codecvt()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>codecvt&lt;char, char, __mbstate_t&gt;::~codecvt()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>codecvt&lt;char, char, __mbstate_t&gt;::do_out(__mbstate_t&amp; char const*, char const*, char const*&amp;, char*, char*, char*&amp;) const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>codecvt&lt;char, char, __mbstate_t&gt;::do_unshift(__mbstate_t&amp; char*, char*, char*&amp;) const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>codecvt&lt;char, char, __mbstate_t&gt;::do_in(__mbstate_t&amp;,</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the `std::codecvt<char, char, __mbstate_t>` class is described by Table 16-298.

**Table 16-298 typeinfo for codecvt<char, char, __mbstate_t>**

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabi::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for codecvt&lt;char, char, __mbstate_t&gt;</td>
</tr>
</tbody>
</table>

**16.1.104.2 Interfaces for Class class codecvt<char, char, __mbstate_t>**

An LSB conforming implementation shall provide the generic methods for Class `std::codecvt<char, char, __mbstate_t>` specified in Table 16-299, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-299 libstdcxx - Class codecvt<char, char, __mbstate_t> Function Interfaces**

<table>
<thead>
<tr>
<th>Function Name</th>
<th>[GLIBCXX_3.4]</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>do_unshift(__mbstate_t&amp;, char*, char*, char*&amp;)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>do_encoding()</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>do_max_length()</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>do_always_noconv()</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>do_in(__mbstate_t&amp;, char const*, char const*, char const*, char const*&amp;, char*, char*&amp;)</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>do_out(__mbstate_t&amp;</code>, char const*, char const*, char const*, char const*&amp;, char*, char*&amp;)`</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>~codecvt()</code></td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::codecvt<char, char, __mbstate_t>` specified in Table 16-300, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-300 libstdcxx - Class codecvt<char, char, __mbstate_t> Data Interfaces**

<table>
<thead>
<tr>
<th>Function Name</th>
<th>[GLIBCXX_3.4]</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>id()</code></td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td><code>typeinfo for codecvt&lt;char, char, __mbstate_t&gt;(GLIBCXX_3.4)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td><code>typeinfo name for codecvt&lt;char, char, __mbstate_t&gt;(GLIBCXX_3.4)</code></td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>
### 16.1.105 Class `codecvt<wchar_t, char, __mbstate_t>`

#### 16.1.105.1 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 16-301.

#### Table 16-301 Primary vtable for `codecvt<wchar_t, char, __mbstate_t>`

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**RTTI**
- `typeinfo 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]:**
- See architecture specific part.

**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 16-302.

#### Table 16-302 typeinfo for `codecvt<wchar_t, char, __mbstate_t>`

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>Vtable for __cxaabi_v1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>typeinfo name for codecvt&lt;wchar_t, char, __mbstate_t&gt;</code></td>
</tr>
</tbody>
</table>

### 16.1.105.2 Interfaces for Class `codecvt<wchar_t, char, __mbstate_t>`

An LSB conforming implementation shall provide the generic methods for Class `std::codecvt<wchar_t, char, __mbstate_t>` specified in Table 16-303, with the full mandatory functionality as described in the referenced underlying specification.
An LSB conforming implementation shall provide the generic data interfaces for Class `std::codecvt<wchar_t, char, __mbstate_t>` specified in Table 16-304, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Virtual Table</th>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]</th>
<th>vfunc[1]</th>
<th>vfunc[2]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 16-305 Primary vtable for `codecvt_byname<char, char, __mbstate_t>`

16.1.106 Class `codecvt_byname<char, char, __mbstate_t>`

16.1.106.1 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 16-305.
The Run Time Type Information for the std::codecvt_byname<char, char, __mbstate_t> class is described by Table 16-306.

### Table 16-306 typeinfo for codecvt_byname<char, char, __mbstate_t>

<table>
<thead>
<tr>
<th>Name</th>
<th>vtable for __cxxabi_v1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>typeinfo name for codecvt_byname&lt;char, char, __mbstate_t&gt;</td>
</tr>
</tbody>
</table>

### 16.1.106.2 Interfaces for Class codecvt_byname<char, char, __mbstate_t> |

An LSB conforming implementation shall provide the generic data interfaces for Class std::codecvt_byname<char, char, __mbstate_t> specified in Table 16-308, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-307 libstdcxx - Class codecvt_byname<char, char, __mbstate_t> Function Interfaces

- `codecvt_byname<char, char, __mbstate_t>::~codecvt_byname()` (GLIBCXX_3.4) [ISOCXX]
- `codecvt_byname<char, char, __mbstate_t>::~codecvt_byname()` (GLIBCXX_3.4) [ISOCXX]
- `codecvt_byname<char, char, __mbstate_t>::~codecvt_byname()` (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class std::codecvt_byname<char, char, __mbstate_t> specified in Table 16-308, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-308 libstdcxx - Class codecvt_byname<char, char, __mbstate_t> Data Interfaces

- typeinfo for codecvt_byname<char, char, __mbstate_t>(GLIBCXX_3.4) [CXXABI-L86]
- typeinfo name for codecvt_byname<char, char, __mbstate_t>(GLIBCXX_3.4) [CXXABI-L86]
16 Libraries

16.1.107 Class codecvt_byname<wchar_t, char, __mbstate_t>

16.1.107.1 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 16-309.

Table 16-309 Primary vtable for codecvt_byname<wchar_t, char, __mbstate_t>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for codecvt_byname&lt;wchar_t, char, __mbstate_t&gt;</td>
</tr>
</tbody>
</table>

| vfunc[0]: | codecvt_byname<wchar_t, char, __mbstate_t>::~codecvt_byname() |
| vfunc[1]: | codecvt_byname<wchar_t, char, __mbstate_t>::~codecvt_byname() |
| 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]: | See architecture specific part. |
| vfunc[8]: | codecvt<wchar_t, char, __mbstate_t>::do_max_length() const |

The Run Time Type Information for the std::codecvt_byname<wchar_t, char, __mbstate_t> class is described by Table 16-310.

Table 16-310 typeinfo for codecvt_byname<wchar_t, char, __mbstate_t>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for codecvt_byname&lt;wchar_t, char, __mbstate_t&gt;</td>
</tr>
</tbody>
</table>
16.1.107.2 Interfaces for Class `codecvt_byname<wchar_t, char, __mbstate_t>`

An LSB conforming implementation shall provide the generic methods for Class `std::codecvt_byname<wchar_t, char, __mbstate_t>` specified in Table 16-311, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>codecvt_byname&lt;wchar_t, char, __mbstate_t&gt;::~codecvt_byname()</code></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::codecvt_byname<wchar_t, char, __mbstate_t>` specified in Table 16-312, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>typeinfo for codecvt_byname&lt;wchar_t, char, __mbstate_t&gt;()</code></td>
</tr>
</tbody>
</table>

16.1.108 Class `collate<char>`

16.1.108.1 Class data for `collate<char>`

The virtual table for the `std::collate<char>` class is described by Table 16-313.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]:</th>
<th>vfunc[1]:</th>
<th>vfunc[2]:</th>
<th>vfunc[3]:</th>
<th>vfunc[4]:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td><code>collate&lt;char&gt;::~collate()</code></td>
<td><code>collate&lt;char&gt;::~collate()</code></td>
<td><code>collate&lt;char&gt;::do_compare(char const*, char const*, char const*, char const*)</code> const</td>
<td><code>collate&lt;char&gt;::do_hash(char const*, char const*)</code> const</td>
<td><code>collate&lt;char&gt;::do_transform(char const*, char const*)</code> const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the `std::collate<char>` class is described by Table 16-314.

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for</th>
</tr>
</thead>
</table>
16.1.108.2 Interfaces for Class `collate<char>`

An LSB conforming implementation shall provide the generic methods for Class `std::collate<char>` specified in Table 16-315, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>::_M_compare</code></td>
<td><code>collate&lt;char&gt;::_M_compare(char const*, char const*) const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>::do_compare</code></td>
<td><code>collate&lt;char&gt;::do_compare(char const*, char const*, char const*, char const*)</code></td>
</tr>
<tr>
<td><code>::do_transform</code></td>
<td><code>collate&lt;char&gt;::do_transform(char const*, char const*) const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>::hash</code></td>
<td><code>collate&lt;char&gt;::hash(char const*, char const*) const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>::compare</code></td>
<td><code>collate&lt;char&gt;::compare(char const*, char const*, char const*, char const*) const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>::do_hash</code></td>
<td><code>collate&lt;char&gt;::do_hash(char const*, char const*) const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>::transform</code></td>
<td><code>collate&lt;char&gt;::transform(char const*, char const*) const(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>~collate</code></td>
<td><code>collate&lt;char&gt;::~collate();</code></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::collate<char>` specified in Table 16-316, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>::id</code></td>
<td><code>collate&lt;char&gt;::id(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>::typeinfo</code></td>
<td><code>typeinfo for collate&lt;char&gt;(GLIBCXX_3.4)</code></td>
</tr>
<tr>
<td><code>::vtable</code></td>
<td><code>vtable for collate&lt;char&gt;(GLIBCXX_3.4)</code></td>
</tr>
</tbody>
</table>

16.1.109 Class `collate<wchar_t>`

16.1.109.1 Class data for `collate<wchar_t>`

The virtual table for the std::collate<wchar_t> class is described by Table 16-317.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>RTTI</td>
</tr>
<tr>
<td>0</td>
<td><code>::typeinfo</code></td>
</tr>
<tr>
<td>0</td>
<td><code>collate&lt;wchar_t&gt;::~collate()</code></td>
</tr>
<tr>
<td>0</td>
<td><code>collate&lt;wchar_t&gt;::~collate()</code></td>
</tr>
<tr>
<td>0</td>
<td><code>collate&lt;wchar_t&gt;::do_compare(wchar_t const*, wchar_t const*, wchar_t const*, wchar_t const*) const</code></td>
</tr>
</tbody>
</table>
The Run Time Type Information for the std::collate<wchar_t> class is described by Table 16-318.

### Table 16-318 typeinfo for collate<wchar_t>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for collate&lt;wchar_t&gt;</td>
</tr>
</tbody>
</table>

### 16.1.109.2 Interfaces for Class collate<wchar_t>

An LSB conforming implementation shall provide the generic methods for Class std::collate<wchar_t> specified in Table 16-319, with the full mandatory functionality as described in the referenced underlying specification.

#### Table 16-319 libstdcxx - Class collate<wchar_t> Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>collate&lt;wchar_t&gt;::_M_compare(wchar_t const*, wchar_t const*)</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>collate&lt;wchar_t&gt;::do_compare(wchar_t const*, wchar_t const*, wchar_t const*, wchar_t const*) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>collate&lt;wchar_t&gt;::do_transform(wchar_t const*, wchar_t const*)</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>collate&lt;wchar_t&gt;::hash(wchar_t const*, wchar_t const*) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>collate&lt;wchar_t&gt;::compare(wchar_t const*, wchar_t const*, wchar_t const*, wchar_t const*) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>collate&lt;wchar_t&gt;::do_hash(wchar_t const*, wchar_t const*) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>collate&lt;wchar_t&gt;::transform(wchar_t const*, wchar_t const*) const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>collate&lt;wchar_t&gt;::~collate()</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>collate&lt;wchar_t&gt;::~collate()</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>collate&lt;wchar_t&gt;::~collate()</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::collate<wchar_t> specified in Table 16-320, with the full mandatory functionality as described in the referenced underlying specification.

#### Table 16-320 libstdcxx - Class collate<wchar_t> Data Interfaces

<table>
<thead>
<tr>
<th>Data Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_guard variable for collate&lt;wchar_t&gt;::id(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>collate&lt;wchar_t&gt;::id(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>typeinfo for collate&lt;wchar_t&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for collate&lt;wchar_t&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for collate&lt;wchar_t&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
16.1.110 Class collate_byname<char>

16.1.110.1 Class data for collate_byname<char>

The virtual table for the std::collate_byname<char> class is described by Table 16-321.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>typeinfo for collate_byname&lt;char&gt;</td>
<td>collate_byname&lt;char&gt;::~collate_byname()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>vfunc[1]: collate_byname&lt;char&gt;::~collate_byname()</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>vfunc[2]: collate&lt;char&gt;::do_compare(char const*, char const*, char const*, char const*) const</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>vfunc[3]: collate&lt;char&gt;::do_transform(char const*, char const*) const</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>vfunc[4]: collate&lt;char&gt;::do_hash(char const*, char const*) const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::collate_byname<char> class is described by Table 16-322.

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxaabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>typeinfo name for collate_byname&lt;char&gt;</td>
</tr>
</tbody>
</table>

16.1.110.2 Interfaces for Class collate_byname<char>

An LSB conforming implementation shall provide the generic methods for Class std::collate_byname<char> specified in Table 16-323, with the full mandatory functionality as described in the referenced underlying specification.

| collate_byname<char>::~collate_byname() (GLIBCXX_3.4) | [ISOCXX] |
| collate_byname<char>::~collate_byname() (GLIBCXX_3.4) | [ISOCXX] |
| collate_byname<char>::~collate_byname() (GLIBCXX_3.4) | [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::collate_byname<char> specified in Table 16-324, with the full mandatory functionality as described in the referenced underlying specification.

| typeinfo for collate_byname<char>(GLIBCXX_3.4) | [CXXABI-1.86] |
| typeinfo name for collate_byname<char>(GLIBCXX_3.4) | [CXXABI-1.86] |
| vtable for collate_byname<char>(GLIBCXX_3.4) | [CXXABI-1.86] |
16.1.111 Class \texttt{collate\_byname\textless wchar\_t\textgreater}

16.1.111.1 Class data for \texttt{collate\_byname\textless wchar\_t\textgreater}

The virtual table for the \texttt{std::collate\_byname\textless wchar\_t\textgreater} class is described by Table 16-325.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>\texttt{typeinfo for \textless \texttt{collate_byname\textless wchar_t\textgreater}}</td>
</tr>
<tr>
<td>vf\texttt{[0]}:</td>
<td>\texttt{collate_byname\textless wchar_t\textgreater::\texttt{~collate_byname\textless wchar_t\textgreater}}</td>
</tr>
<tr>
<td>vf\texttt{[1]}:</td>
<td>\texttt{collate_byname\textless wchar_t\textgreater::\texttt{~collate_byname\textless wchar_t\textgreater}}</td>
</tr>
<tr>
<td>vf\texttt{[2]}:</td>
<td>\texttt{collate\textless wchar_t\textgreater::do_compare(wchar_t const*, wchar_t const*, wchar_t const*, wchar_t const*) const}</td>
</tr>
<tr>
<td>vf\texttt{[3]}:</td>
<td>\texttt{collate\textless wchar_t\textgreater::do_transform(wchar_t const*, wchar_t const*) const}</td>
</tr>
<tr>
<td>vf\texttt{[4]}:</td>
<td>\texttt{collate\textless wchar_t\textgreater::do_hash(wchar_t const*, wchar_t const*) const}</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the \texttt{std::collate\_byname\textless wchar\_t\textgreater} class is described by Table 16-326.

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>\texttt{__cxxabi___si_class_type_info}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>\texttt{typeinfo name for \textless \texttt{collate_byname\textless wchar_t\textgreater}}</td>
</tr>
</tbody>
</table>

16.1.111.2 Interfaces for Class \texttt{collate\_byname\textless wchar\_t\textgreater}

An LSB conforming implementation shall provide the generic methods for Class \texttt{std::collate\_byname\textless wchar\_t\textgreater} specified in Table 16-327, with the full mandatory functionality as described in the referenced underlying specification.

| \texttt{collate\_byname\textless wchar\_t\textgreater::\texttt{~collate\_byname\textless wchar\_t\textgreater}}(GLIBCXX\_3.4) [ISO\_CXX] |
| \texttt{collate\_byname\textless wchar\_t\textgreater::\texttt{~collate\_byname\textless wchar\_t\textgreater}}(GLIBCXX\_3.4) [ISO\_CXX] |
| \texttt{collate\_byname\textless wchar\_t\textgreater::\texttt{~collate\_byname\textless wchar\_t\textgreater}}(GLIBCXX\_3.4) [ISO\_CXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class \texttt{std::collate\_byname\textless wchar\_t\textgreater} specified in Table 16-328, with the full mandatory functionality as described in the referenced underlying specification.

| typeinfo for \texttt{collate\_byname\textless wchar\_t\textgreater}(GLIBCXX\_3.4) [CXX\_ABI-1.86] |
| typeinfo name for \texttt{collate\_byname\textless wchar\_t\textgreater}(GLIBCXX\_3.4) [CXX\_ABI-1.86] |
| vtable for \texttt{collate\_byname\textless wchar\_t\textgreater}(GLIBCXX\_3.4) [CXX\_ABI-1.86] |
16.1.112 Class time_base

16.1.112.1 Class data for time_base

The Run Time Type Information for the std::time_base class is described by Table 16-329.

Table 16-329 typeinfo for time_base

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for time_base</td>
</tr>
</tbody>
</table>

16.1.112.2 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 relevant architecture specific part of this specification.

An LSB conforming implementation shall provide the generic data interfaces for Class std::time_base specified in Table 16-330, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-330 libstdcxx - Class time_base Data Interfaces

<table>
<thead>
<tr>
<th>typeinfo name for time_base(GLIBCXX_3.4) [CXXABI-1.86]</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo name for time_base(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.113 Class time_get_byname<char, istreambuf_iterator<char, char_traits<char> > >

16.1.113.1 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 by Table 16-331.

Table 16-331 Primary vtable for time_get_byname<char, istreambuf_iterator<char, char_traits<char> > >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; ::~time_get_byname()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; ::~time_get_byname()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>time_get&lt;char&gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; ::do_date_order() const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>time_get&lt;char&gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
</tr>
</tbody>
</table>
### 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 16-332

#### Table 16-332 typeinfo for time_get_byname<char, istreambuf_iterator<char, char_traits<char> > >

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;</td>
</tr>
</tbody>
</table>

#### 16.1.113.2 Interfaces for Class time_get_byname<char, istreambuf_iterator<char, char_traits<char> > > >

An LSB conforming implementation shall provide the generic methods for Class std::time_get_byname<char, std::istreambuf_iterator<char, std::char_traits<char> > > >
specified in Table 16-333, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-333 libstdcxx - Class time_get_byname<char, istreambuf_iterator<char, char_traits<char> > > Function Interfaces

<table>
<thead>
<tr>
<th>Function Name</th>
<th>GLIBCXX Version</th>
<th>ISOCXX Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; ::~time_get_byname()</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; ::~time_get_byname()</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; ::~time_get_byname()</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::time_get_byname<char, std::istreambuf_iterator<char, std::char_traits<char> > > specified in Table 16-334, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-334 libstdcxx - Class time_get_byname<char, istreambuf_iterator<char, char_traits<char> > > Data Interfaces

<table>
<thead>
<tr>
<th>Interface Name</th>
<th>GLIBCXX Version</th>
<th>ISOCXX Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeid for time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>vtable name for time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>vfunc[0]: time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; ::~time_get_byname()</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>vfunc[1]: time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; ::~time_get_byname()</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
</tbody>
</table>

16.1.114 Class time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

16.1.114.1 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 by Table 16-335.

Table 16-335 Primary vtable for time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeid for time_get_byname&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>time_get_byname&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::~time_get_byname()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>time_get_byname&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::~time_get_byname()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::~time_get_byname()</td>
</tr>
</tbody>
</table>
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 16-336.

<table>
<thead>
<tr>
<th>Name</th>
<th>vtable for <code>__cxxabiv1::__si_class_type_info</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>time_get&lt;wchar_t&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>::do_date_order()</code> const</td>
<td></td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td></td>
</tr>
</tbody>
</table>
| `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>
  ::do_get_time(istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, tm*) const` |                                               |
| vfunc[4]:                                      |                                               |
| `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>
  ::do_get_date(istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, tm*) const` |                                               |
| vfunc[5]:                                      |                                               |
| `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>
  ::do_get_weekday(istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, tm*) const` |                                               |
| vfunc[6]:                                      |                                               |
| `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>
  ::do_get_monthname(istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, tm*) const` |                                               |
| vfunc[7]:                                      |                                               |
| `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>
  ::do_get_year(istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, tm*) const` |                                               |

Table 16-336 typeinfo for `time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>`
16 Libraries

LSB Core - Generic 5.0

16.1.114.2 Interfaces for Class time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the generic methods for Class std::time_get_byname<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 16-337, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-337 libstdcxx - Class time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > > > Function Interfaces

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]</th>
<th>vfunc[1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_get_byname&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::~time_get_byname()</td>
<td>379</td>
<td>0</td>
<td>(GLIBCXX_3.4)</td>
<td>time_get_byname&lt;wchar_t, std::istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::~time_get_byname()</td>
<td>time_get_byname&lt;wchar_t, std::istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::~time_get_byname()</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::time_get_byname<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 16-338, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-338 libstdcxx - Class time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > > > Data Interfaces

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]</th>
<th>vfunc[1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>typeinfo for time_get_byname&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; (GLIBCXX_3.4)</td>
<td>379</td>
<td>0</td>
<td>(GLIBCXX_3.4)</td>
<td>time_get_byname&lt;wchar_t, std::istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::~time_get_byname()</td>
<td>time_get_byname&lt;wchar_t, std::istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::~time_get_byname()</td>
</tr>
</tbody>
</table>

16.1.115 Class time_put_byname<char, ostreambuf_iterator<char, char_traits<char> > >

16.1.115.1 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 by Table 16-339.

Table 16-339 Primary vtable for time_put_byname<char, ostreambuf_iterator<char, char_traits<char> > > >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]</th>
<th>vfunc[1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>typeinfo for time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;</td>
<td>time_put_byname&lt;char, std::ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; ::~time_put_byname()</td>
<td>time_put_byname&lt;char, std::ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; ::~time_put_byname()</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the `std::time_put_byname<char, ostreambuf_iterator<char, char_traits<char>> >` class is described by Table 16-340.

### Table 16-340 typeinfo for `time_put_byname<char, ostreambuf_iterator<char, char_traits<char>> >` > >

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt; &gt;</td>
</tr>
</tbody>
</table>

#### 16.1.115.2 Interfaces for Class `time_put_byname<char, ostreambuf_iterator<char, char_traits<char>> >` > >

An LSB conforming implementation shall provide the generic methods for Class `std::time_put_byname<char, std::ostreambuf_iterator<char, std::char_traits<char>> >` > > specified in Table 16-341, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-341 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() | GLIBCXX_3.4 [ISOCXX] |
| time_put_byname<char, ostreambuf_iterator<char, char_traits<char>> > > >::~time_put_byname() | GLIBCXX_3.4 [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class `std::time_put_byname<char, std::ostreambuf_iterator<char, std::char_traits<char>> >` > > specified in Table 16-342, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-342 libstdcxx - Class `time_put_byname<char, ostreambuf_iterator<char, char_traits<char>> >` > > Data Interfaces

| typeinfo for time_put_byname<char, ostreambuf_iterator<char, char_traits<char>> > > > | GLIBCXX_3.4 [CXXABI-1.86] |
| typeinfo name for time_put_byname<char, ostreambuf_iterator<char, char_traits<char>> > > > | GLIBCXX_3.4 [CXXABI-1.86] |
| vtable for time_put_byname<char, ostreambuf_iterator<char, char_traits<char>> > > > | GLIBCXX_3.4 [CXXABI-1.86] |
16.1.116 Class time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

16.1.116.1 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 by Table 16-343.

Table 16-343 Primary vtable for time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

| Base Offset | 0 |
| Virtual Base Offset | 0 |
| RTTI | typeinfo for time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > |
| vfunc[0]: | time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > >::time_put_byname() |
| vfunc[1]: | time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > >::time_put_byname() |
| vfunc[2]: | time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > >::do_put(ostreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, wchar_t const*, char, char) const |

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 16-344.

Table 16-344 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> > > > |

16.1.116.2 Interfaces for Class time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the generic methods for Class std::time_put_byname<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > > specified in Table 16-345, with the full mandatory functionality as described in the referenced underlying specification.
An LSB conforming implementation shall provide the generic data interfaces for Class std::time_put_byname<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 16-346, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-346 libstdcxx - Class time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > Data Interfaces

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]:</th>
<th>vfunc[1]:</th>
<th>vfunc[2]:</th>
<th>vfunc[3]:</th>
</tr>
</thead>
</table>
| 0           | 0                   |      | time_get<
|             |                     |      | char,     | char,     | time_get<
|             |                     |      | istreambuf_iterator<char, | istreambuf_iterator<char, | char,     | char,     | char> >> |
|             |                     |      | char_traits<char> > >> | char_traits<char> > >> | time_get() | time_get() |
|             |                     |      | do_date_order() const    | do_date_order() const    |           |           |
|             |                     |      | do_get_time(istreambuf_iterator<char, | do_get_time(istreambuf_iterator<char, | char_traits<char> > > | char_traits<char> > > |

16.1.117 Class time_get<char, istreambuf_iterator<char, char_traits<char> > > >>

16.1.117.1 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 by Table 16-347.

### Table 16-347 Primary vtable for time_get<char, istreambuf_iterator<char, char_traits<char> > > >>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]:</th>
<th>vfunc[1]:</th>
<th>vfunc[2]:</th>
<th>vfunc[3]:</th>
</tr>
</thead>
</table>
| 0           | 0                   |      | time_get<
|             |                     |      | char,     | char,     | time_get<
|             |                     |      | istreambuf_iterator<char, | istreambuf_iterator<char, | char,     | char,     | char> >> |
|             |                     |      | char_traits<char> > >> | char_traits<char> > >> | time_get() | time_get() |
|             |                     |      | do_date_order() const    | do_date_order() const    |           |           |
|             |                     |      | do_get_time(istreambuf_iterator<char, | do_get_time(istreambuf_iterator<char, | char_traits<char> > > | char_traits<char> > > |

© 2015 Linux Foundation
16.1.117.2 Interfaces for Class `time_get<char, istreambuf_iterator<char, char_traits<char>>>`

An LSB conforming implementation shall provide the generic methods for Class std::time_get<char, std::istreambuf_iterator<char, std::char_traits<char>>> specified in Table 16-348, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-348 libstdcxx - Class `time_get<char, istreambuf_iterator<char, char_traits<char>>>` > > Function Interfaces

<table>
<thead>
<tr>
<th>Function Iterator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::date_order()</code></td>
<td>const(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td><code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::do_get_date()</code></td>
<td>const(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td><code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::do_get_weekday()</code></td>
<td>const(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td><code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::do_get_monthname()</code></td>
<td>const(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
<tr>
<td><code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::do_get_year()</code></td>
<td>const(GLIBCXX_3.4) [ISOcxx]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class `std::time_get<char, std::istreambuf_iterator<char, std::char_traits<char> > >` specified in Table 16-349, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-349  libstdcxx - Class `time_get<char, std::istreambuf_iterator<char, std::char_traits<char> > >` Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>::do_get_year</code></td>
<td>Retrieves the year from the input stream.</td>
</tr>
<tr>
<td><code>::do_get_weekday</code></td>
<td>Retrieves the weekday from the input stream.</td>
</tr>
<tr>
<td><code>::do_get_monthname</code></td>
<td>Retrieves the month name from the input stream.</td>
</tr>
<tr>
<td><code>::get_date</code></td>
<td>Retrieves the date from the input stream.</td>
</tr>
<tr>
<td><code>::get_time</code></td>
<td>Retrieves the time from the input stream.</td>
</tr>
<tr>
<td><code>::get_year</code></td>
<td>Retrieves the year from the input stream.</td>
</tr>
<tr>
<td><code>::get_monthname</code></td>
<td>Retrieves the month name from the input stream.</td>
</tr>
<tr>
<td><code>::get_weekday</code></td>
<td>Retrieves the weekday from the input stream.</td>
</tr>
<tr>
<td><code>::_M_extract_via_format</code></td>
<td>Extracts data from the input stream using a format.</td>
</tr>
<tr>
<td><code>::~time_get</code></td>
<td>Destroys the time_get object.</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation

723
16.1.118 Class `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >` 

16.1.118.1 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 by Table 16-350.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::~time_get()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::do_date_order() const</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::do_get_time(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, tm*) const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::do_get_date(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, tm*) const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::do_get_time(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, tm*) const</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::do_get_date(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, tm*) const</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic methods for Class std::time_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > > specified in Table 16-351, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-351 libstdcxx - Class time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > > Function Interfaces

```
<table>
<thead>
<tr>
<th>Function</th>
<th>Declaration</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;::date_order()</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;::do_date_order(time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; &amp; )</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;::do_year(time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; &amp; )</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;::get_weekday(time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; &amp; )</td>
<td>const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>
```
An LSB conforming implementation shall provide the generic data interfaces for Class
std::time_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 16-352, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 16-352 libstdcxx - Class time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; Data Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>guard variable for time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::id(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::id(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::id(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; ::id(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo for time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; (GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; (GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; (GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>
16.1.119 Class `time_put<char, ostreambuf_iterator<char, char_traits<char> > >` > >

16.1.119.1 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 by Table 16-353.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td><code>typeid for time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</code></td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td><code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::~time_put()</code></td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td><code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::~time_put()</code></td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td><code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, char, tm const*, char, char) const</code></td>
</tr>
</tbody>
</table>

The Run Time Type Information for the `std::time_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > >` class is described by Table 16-354.

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td><code>typeid for time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</code></td>
</tr>
<tr>
<td>flags:</td>
<td>8</td>
</tr>
<tr>
<td>basetype:</td>
<td><code>typeid for locale::facet</code> 2</td>
</tr>
<tr>
<td>basetype:</td>
<td><code>typeid for time_base</code> 2</td>
</tr>
</tbody>
</table>

16.1.119.2 Interfaces for Class `time_put<char, ostreambuf_iterator<char, char_traits<char> > >` > >

An LSB conforming implementation shall provide the generic methods for Class `std::time_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > >` > > specified in Table 16-355, with the full mandatory functionality as described in the referenced underlying specification.
16 Libraries

LSB Core - Generic 5.0

Table 16-355  libstdcxx - Class time_put<char, ostreambuf_iterator<char, char_traits<char> > > > Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;, ios_base&amp;, char, tm const*, char const*, char const*) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>do_put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;, ios_base&amp;, char, tm const*, char, char) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>~time_put() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::time_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > specified in Table 16-356, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-356  libstdcxx - Class time_put<char, ostreambuf_iterator<char, char_traits<char> > > > Data Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guard variable for time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;</td>
<td>id(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>id(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>typeinfo for time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.120 Class time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

16.1.120.1 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 by Table 16-357

Table 16-357 Primary vtable for time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</td>
</tr>
</tbody>
</table>
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 16-358.

Table 16-358 typeinfo for time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > class is described by Table 16-358

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabi__si_class_t&lt;__si_type_info&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for __temp_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</td>
</tr>
<tr>
<td>flags:</td>
<td>8</td>
</tr>
<tr>
<td>basetype:</td>
<td>typeinfo for locale::facet 2</td>
</tr>
<tr>
<td>basetype:</td>
<td>typeinfo for time_base 2</td>
</tr>
</tbody>
</table>

16.1.120.2 Interfaces for Class time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the generic methods for Class std::time_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > > specified in Table 16-359, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-359 libstdcxx - Class time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > > Function Interfaces

| time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > |
| :>put(ostreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, wchar_t, tm const*, wchar_t const*, wchar_t const*) const(GLIBCXX_3.4) [ISO<XXXX]] |
| time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > |
| :>put(ostreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, wchar_t, tm const*, char, char) const(GLIBCXX_3.4) [ISO<XXXX]] |
| time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > |
| :>do_put(ostreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, wchar_t, tm const*, char, char) const(GLIBCXX_3.4) [ISO<XXXX]] |
| time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > |
| :>~time_put(GLIBCXX_3.4) [ISO<XXXX]] |
| time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > |
| :>~time_put(GLIBCXX_3.4) [ISO<XXXX]] |
An LSB conforming implementation shall provide the generic data interfaces for Class `std::time_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > >` specified in Table 16-360, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-360  libstdcxx - Class `time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >` Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guard variable for <code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code></td>
<td><code>::id(GLIBCXX_3.4)</code> [CXXABI-1.86]</td>
</tr>
<tr>
<td><code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code></td>
<td><code>::id(GLIBCXX_3.4)</code> [ISOCXX]</td>
</tr>
<tr>
<td>Typeinfo for <code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code></td>
<td><code>&gt;(GLIBCXX_3.4)</code> [CXXABI-1.86]</td>
</tr>
<tr>
<td>Typeinfo name for <code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code></td>
<td><code>&gt;(GLIBCXX_3.4)</code> [CXXABI-1.86]</td>
</tr>
<tr>
<td>Vtable for <code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code></td>
<td><code>&gt;(GLIBCXX_3.4)</code> [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.121 Class `moneypunct<char, false>`

16.1.121.1 Class data for `moneypunct<char, false>`

The virtual table for the `std::moneypunct<char, false>` class is described by Table 16-361.

Table 16-361 Primary vtable for `moneypunct<char, false>`

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td><code>moneypunct&lt;char, false&gt;::~moneypunct()</code></td>
<td><code>moneypunct&lt;char, false&gt;::~moneypunct()</code></td>
<td><code>moneypunct&lt;char, false&gt;::do_decimal_point() const</code></td>
<td><code>moneypunct&lt;char, false&gt;::do_thousands_sep() const</code></td>
<td><code>moneypunct&lt;char, false&gt;::do_grouping() const</code></td>
<td><code>moneypunct&lt;char, false&gt;::do_curr_symbol() const</code></td>
<td><code>moneypunct&lt;char, false&gt;::do_pos_format() const</code></td>
<td><code>moneypunct&lt;char, false&gt;::do_negative_sign() const</code></td>
<td><code>moneypunct&lt;char, false&gt;::do_frac_digits() const</code></td>
<td><code>moneypunct&lt;char, false&gt;::do_pos_format() const</code></td>
<td><code>moneypunct&lt;char, false&gt;::do_negative_sign() const</code></td>
</tr>
</tbody>
</table>
16.1.121.2 Interfaces for Class moneypunct<char, false>

An LSB conforming implementation shall provide the generic methods for Class std::moneypunct<char, false> specified in Table 16-362, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-362 libstdcxx - Class moneypunct<char, false> Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>moneypunct&lt;char, false&gt;::neg_format() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::pos_format() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::curr_symbol() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::do_grouping() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::frac_digits() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::decimal_point() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::do_neg_format() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::do_pos_format() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::negative_sign() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::positive_sign() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::thousands_sep() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::do_curr_symbol() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::do_frac_digits() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::do_decimal_point() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::do_negative_sign() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::do_positive_sign() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::do_thousands_sep() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::grouping() const</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::moneypunct<char, false> specified in Table 16-363, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-363 libstdcxx - Class moneypunct<char, false> Data Interfaces

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>guard variable for moneypunct&lt;char, false&gt;::id(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::id(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, false&gt;::intl(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4 [ISOCXX]</td>
</tr>
<tr>
<td>typeinfo for moneypunct&lt;char, false&gt;(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for moneypunct&lt;char, false&gt;(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for moneypunct&lt;char, false&gt;(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
16.1.122 Class moneypunct<char, true>

16.1.122.1 Class data for moneypunct<char, true>

The virtual table for the std::moneypunct<char, true> class is described by Table 16-364.

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for moneypunct&lt;char, true&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>moneypunct&lt;char, true&gt;::~moneypunct()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>moneypunct&lt;char, true&gt;::~moneypunct()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>moneypunct&lt;char, true&gt;::do_decimal_point() const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>moneypunct&lt;char, true&gt;::do_thousands_sep() const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>moneypunct&lt;char, true&gt;::do_grouping() const</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>moneypunct&lt;char, true&gt;::do_curr_symbol() const</td>
</tr>
<tr>
<td>vfunc[6]:</td>
<td>moneypunct&lt;char, true&gt;::do_positive_sign() const</td>
</tr>
<tr>
<td>vfunc[7]:</td>
<td>moneypunct&lt;char, true&gt;::do_negative_sign() const</td>
</tr>
<tr>
<td>vfunc[8]:</td>
<td>moneypunct&lt;char, true&gt;::do_frac_digits() const</td>
</tr>
<tr>
<td>vfunc[9]:</td>
<td>moneypunct&lt;char, true&gt;::do_pos_format() const</td>
</tr>
<tr>
<td>vfunc[10]:</td>
<td>moneypunct&lt;char, true&gt;::do_neg_format() const</td>
</tr>
</tbody>
</table>

16.1.122.2 Interfaces for Class moneypunct<char, true>

An LSB conforming implementation shall provide the generic methods for Class std::moneypunct<char, true> specified in Table 16-365, with the full mandatory functionality as described in the referenced underlying specification.

| moneypunct<char, true>::neg_format() const(GLIBCXX_3.4) [ISOCXX] |
| moneypunct<char, true>::pos_format() const(GLIBCXX_3.4) [ISOCXX] |
| moneypunct<char, true>::curr_symbol() const(GLIBCXX_3.4) [ISOCXX] |
| moneypunct<char, true>::do_grouping() const(GLIBCXX_3.4) [ISOCXX] |
| moneypunct<char, true>::frac_digits() const(GLIBCXX_3.4) [ISOCXX] |
| moneypunct<char, true>::decimal_point() const(GLIBCXX_3.4) [ISOCXX] |
| moneypunct<char, true>::do_neg_format() const(GLIBCXX_3.4) [ISOCXX] |
| moneypunct<char, true>::do_pos_format() const(GLIBCXX_3.4) [ISOCXX] |
| moneypunct<char, true>::negative_sign() const(GLIBCXX_3.4) [ISOCXX] |
| moneypunct<char, true>::positive_sign() const(GLIBCXX_3.4) [ISOCXX] |
| moneypunct<char, true>::thousands_sep() const(GLIBCXX_3.4) [ISOCXX] |
An LSB conforming implementation shall provide the generic data interfaces for Class std::moneypunct<char, true> specified in Table 16-366, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-366 libstdcxx - Class moneypunct<char, true> Data Interfaces**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guard variable for moneypunct&lt;char, true&gt;::id( GLIBCXX_3.4 )</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>moneypunct&lt;char, true&gt;::id( GLIBCXX_3.4 )</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;char, true&gt;::intl( GLIBCXX_3.4 )</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>typeinfo for moneypunct&lt;char, true&gt;( GLIBCXX_3.4 )</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for moneypunct&lt;char, true&gt;( GLIBCXX_3.4 )</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>

### 16.1.123 Class moneypunct<wchar_t, false>

#### 16.1.123.1 Class data for moneypunct<wchar_t, false>

The virtual table for the std::moneypunct<wchar_t, false> class is described by Table 16-367.

**Table 16-367 Primary vtable for moneypunct<wchar_t, false>**

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for moneypunct&lt;wchar_t, false&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>moneypunct&lt;wchar_t, false&gt;::~moneypunct()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>moneypunct&lt;wchar_t, false&gt;::~moneypunct()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>moneypunct&lt;wchar_t, false&gt;::do_decimal_point() const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>moneypunct&lt;wchar_t, false&gt;::do_thousands_sep() const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>moneypunct&lt;wchar_t, false&gt;::do_grouping() const</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>moneypunct&lt;wchar_t, false&gt;::do_curr_symbol() const</td>
</tr>
</tbody>
</table>
16 Libraries

LSB Core - Generic 5.0

| vfunc[6]: | moneypunct<wchar_t, false>::do_positive_sign() const |
| vfunc[7]: | moneypunct<wchar_t, false>::do_negative_sign() const |
| vfunc[8]: | moneypunct<wchar_t, false>::do_frac_digits() const |
| vfunc[9]: | moneypunct<wchar_t, false>::do_pos_format() const |
| vfunc[10]: | moneypunct<wchar_t, false>::do_neg_format() const |

16.1.123.2 Interfaces for Class moneypunct<wchar_t, false>

An LSB conforming implementation shall provide the generic methods for Class std::moneypunct<wchar_t, false> specified in Table 16-368, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-368 Libstdcxx - Class moneypunct<wchar_t, false> Function Interfaces

| moneypunct<wchar_t, false>::neg_format() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::pos_format() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::curr_symbol() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::do_grouping() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::frac_digits() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::decimal_point() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::do_neg_format() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::do_pos_format() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::negative_sign() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::positive_sign() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::thousands_sep() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::do_curr_symbol() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::do_frac_digits() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::do_decimal_point() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::do_negative_sign() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::do_positive_sign() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::do_thousands_sep() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::grouping() const(GLIBCXX_3.4) | [ISOCXX] |
| moneypunct<wchar_t, false>::_M_initialize_moneypunct(__locale_struct*, char const*)(GLIBCXX_3.4) | [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::moneypunct<wchar_t, false> specified in Table 16-369, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-369 Libstdcxx - Class moneypunct<wchar_t, false> Data Interfaces

| guard variable for moneypunct<wchar_t, false>::id(GLIBCXX_3.4) | [CXXABI-1.86] |
| moneypunct<wchar_t, false>::id(GLIBCXX_3.4) | [ISOCXX] |

© 2015 Linux Foundation
16.1.124 Class `moneypunct<wchar_t, true>`

16.1.124.1 Class data for `moneypunct<wchar_t, true>`

The virtual table for the std::moneypunct<wchar_t, true> class is described by Table 16-370.

Table 16-370 Primary vtable for moneypunct<wchar_t, true>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for moneypunct&lt;wchar_t, true&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::~moneypunct()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::~moneypunct()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_decimal_point() const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_thousands_sep() const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_grouping() const</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_curr_symbol() const</td>
</tr>
<tr>
<td>vfunc[6]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_positive_sign() const</td>
</tr>
<tr>
<td>vfunc[7]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_negative_sign() const</td>
</tr>
<tr>
<td>vfunc[8]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_frac_digits() const</td>
</tr>
<tr>
<td>vfunc[9]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_pos_format() const</td>
</tr>
<tr>
<td>vfunc[10]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_neg_format() const</td>
</tr>
</tbody>
</table>

16.1.124.2 Interfaces for Class `moneypunct<wchar_t, true>`

An LSB conforming implementation shall provide the generic methods for Class std::moneypunct<wchar_t, true> specified in Table 16-371, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-371 libstdc++ - Class moneypunct<wchar_t, true> Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::neg_format() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::pos_format() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::curr_symbol() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_grouping() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::frac_digits() const(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class std::moneypunct<wchar_t, true> specified in Table 16-372, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-372 libstdcxx - Class moneypunct<wchar_t, true> Data Interfaces

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_decimal_point() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>vfunc[0]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_decimal_point() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_neg_format() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>vfunc[1]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_neg_format() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_pos_format() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>vfunc[2]:</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_pos_format() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::negative_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::negative_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::positive_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::positive_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::thousands_sep() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::thousands_sep() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_curr_symbol() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_curr_symbol() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_frac_digits() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_frac_digits() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_decimal_point() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_decimal_point() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_negative_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_negative_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_positive_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_positive_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_negative_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_negative_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_positive_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_positive_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_negative_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_negative_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_positive_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_positive_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_negative_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_negative_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
<tr>
<td>moneypunct&lt;wchar_t, true&gt;::do_positive_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td>moneypunct&lt;wchar_t, true&gt;::do_positive_sign() const(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4</td>
<td>GLIBCXX_3.4</td>
<td></td>
</tr>
</tbody>
</table>

### 16.1.125 Class moneypunct_byname<char, false>

### 16.1.125.1 Class data for moneypunct_byname<char, false>

The virtual table for the std::moneypunct_byname<char, false> class is described by Table 16-373

### Table 16-373 Primary vtable for moneypunct_byname<char, false>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>vfunc[0]:</td>
<td>moneypunct_byname&lt;char, false&gt;::~moneypunct_byname()</td>
<td>moneypunct_byname&lt;char, false&gt;::~moneypunct_byname()</td>
<td>moneypunct&lt;char, false&gt;::do_decimal_point() const</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the std::moneypunct_byname<char, false> class is described by Table 16-374.

### Table 16-374 typeinfo for moneypunct_byname<char, false>

<table>
<thead>
<tr>
<th>Name</th>
<th>typeinfo name for moneypunct_byname&lt;char, false&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Vtable</td>
<td>vtable for __cxxabiv1::__si_class_type_info</td>
</tr>
</tbody>
</table>

### 16.1.125.2 Interfaces for Class moneypunct_byname<char, false>

An LSB conforming implementation shall provide the generic methods for Class std::moneypunct_byname<char, false> specified in Table 16-375, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-375 libstdcxx - Class moneypunct_byname<char, false> Function Interfaces

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>moneypunct_byname&lt;char, false&gt;::~moneypunct_byname()</td>
<td>[GLIBCXX_3.4] [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct_byname&lt;char, false&gt;::~moneypunct_byname()</td>
<td>[GLIBCXX_3.4] [ISOCXX]</td>
</tr>
<tr>
<td>moneypunct_byname&lt;char, false&gt;::~moneypunct_byname()</td>
<td>[GLIBCXX_3.4] [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::moneypunct_byname<char, false> specified in Table 16-376, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-376 libstdcxx - Class moneypunct_byname<char, false> Data Interfaces

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>moneypunct_byname&lt;char, false&gt;::intl(GLIBCXX_3.4)</td>
<td>[ISOCXX]</td>
</tr>
<tr>
<td>typeinfo for moneypunct_byname&lt;char, false&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>typeinfo name for moneypunct_byname&lt;char, false&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for moneypunct_byname&lt;char, false&gt;(GLIBCXX_3.4)</td>
<td>[CXXABI-1.86]</td>
</tr>
</tbody>
</table>
16.1.126 Class moneypunct_byname<char, true>

16.1.126.1 Class data for moneypunct_byname<char, true>

The virtual table for the std::moneypunct_byname<char, true> class is described by Table 16-377.

Table 16-377 Primary vtable for moneypunct_byname<char, true>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for moneypunct_byname&lt;char, true&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>moneypunct_byname&lt;char, true&gt;::~moneypunct_byname()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>moneypunct_byname&lt;char, true&gt;::~moneypunct_byname()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>moneypunct&lt;char, true&gt;::do_decimal_point() const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>moneypunct&lt;char, true&gt;::do_thousands_sep() const</td>
</tr>
<tr>
<td>vfunc[4]:</td>
<td>moneypunct&lt;char, true&gt;::do_grouping() const</td>
</tr>
<tr>
<td>vfunc[5]:</td>
<td>moneypunct&lt;char, true&gt;::do_curr_symbol() const</td>
</tr>
<tr>
<td>vfunc[6]:</td>
<td>moneypunct&lt;char, true&gt;::do_positive_sign() const</td>
</tr>
<tr>
<td>vfunc[7]:</td>
<td>moneypunct&lt;char, true&gt;::do_negative_sign() const</td>
</tr>
<tr>
<td>vfunc[8]:</td>
<td>moneypunct&lt;char, true&gt;::do_frac_digits() const</td>
</tr>
<tr>
<td>vfunc[9]:</td>
<td>moneypunct&lt;char, true&gt;::do_pos_format() const</td>
</tr>
<tr>
<td>vfunc[10]:</td>
<td>moneypunct&lt;char, true&gt;::do_neg_format() const</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::moneypunct_byname<char, true> class is described by Table 16-378.

Table 16-378 typeinfo for moneypunct_byname<char, true>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabi_v1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for moneypunct_byname&lt;char, true&gt;</td>
</tr>
</tbody>
</table>

16.1.126.2 Interfaces for Class moneypunct_byname<char, true>

An LSB conforming implementation shall provide the generic methods for Class std::moneypunct_byname<char, true> specified in Table 16-379, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-379 libstdcxx - Class moneypunct_byname<char, true> Function Interfaces

| moneypunct_byname<char, true>::~moneypunct_byname() (GLIBCXX_3.4) |
An LSB conforming implementation shall provide the generic data interfaces for Class std::moneypunct_byname<char, true> specified in Table 16-380, with the full mandatory functionality as described in the referenced underlying specification.

16.1.127 Class moneypunct_byname<wchar_t, false>

16.1.127.1 Class data for moneypunct_byname<wchar_t, false>

The virtual table for the std::moneypunct_byname<wchar_t, false> class is described by Table 16-381.
The Run Time Type Information for the `std::moneypunct_byname<wchar_t, false>` class is described by Table 16-382.

Table 16-382 typeinfo for moneypunct_byname<wchar_t, false>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for moneypunct_byname&lt;wchar_t, false&gt;</td>
</tr>
</tbody>
</table>

16.1.127.2 Interfaces for Class moneypunct_byname<wchar_t, false>

An LSB conforming implementation shall provide the generic methods for Class `std::moneypunct_byname<wchar_t, false>` specified in Table 16-383, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-383 libstdcxx - Class moneypunct_byname<wchar_t, false> Function Interfaces

| moneypunct_byname<wchar_t, false>::~moneypunct_byname() (GLIBCXX_3.4) | [ISOCXX] |
| moneypunct_byname<wchar_t, false>::~moneypunct_byname() (GLIBCXX_3.4) | [ISOCXX] |
| moneypunct_byname<wchar_t, false>::~moneypunct_byname() (GLIBCXX_3.4) | [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class `std::moneypunct_byname<wchar_t, false>` specified in Table 16-384, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-384 libstdcxx - Class moneypunct_byname<wchar_t, false> Data Interfaces

| moneypunct_byname<wchar_t, false>::intl(GLIBCXX_3.4) | [ISOCXX] |
| typeinfo for moneypunct_byname<wchar_t, false>(GLIBCXX_3.4) | [CXXABI-1.86] |
| typeinfo name for moneypunct_byname<wchar_t, false>(GLIBCXX_3.4) | [CXXABI-1.86] |
| vtable for moneypunct_byname<wchar_t, false>(GLIBCXX_3.4) | [CXXABI-1.86] |

16.1.128 Class moneypunct_byname<wchar_t, true>

16.1.128.1 Class data for moneypunct_byname<wchar_t, true>

The virtual table for the `std::moneypunct_byname<wchar_t, true>` class is described by Table 16-385.

Table 16-385 Primary vtable for moneypunct_byname<wchar_t, true>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for moneypunct_byname&lt;wchar_t, true&gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>moneypunct_byname&lt;wchar_t, true&gt;::~moneypunct_byname()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>moneypunct_byname&lt;wchar_t, true&gt;::~moneypunct_byname()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>moneypunct&lt;wchar_t, true&gt;</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
The Run Time Type Information for the std::moneypunct_byname<wchar_t, true> class is described by Table 16-386.

### Table 16-386 typeinfo for moneypunct_byname<wchar_t, true>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for moneypunct_byname&lt;wchar_t, true&gt;</td>
</tr>
</tbody>
</table>

#### 16.1.128.2 Interfaces for Class moneypunct_byname<wchar_t, true>

An LSB conforming implementation shall provide the generic methods for Class std::moneypunct_byname<wchar_t, true> specified in Table 16-387, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-387 libstdcxx - Class moneypunct_byname<wchar_t, true> Function Interfaces

| moneypunct_byname<wchar_t, true>::~moneypunct_byname()(GLIBCXX_3.4) [ISOcxx] |
| moneypunct_byname<wchar_t, true>::~moneypunct_byname()(GLIBCXX_3.4) [ISOcxx] |
| moneypunct_byname<wchar_t, true>::~moneypunct_byname()(GLIBCXX_3.4) [ISOcxx] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::moneypunct_byname<wchar_t, true> specified in Table 16-388, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-388 libstdcxx - Class moneypunct_byname<wchar_t, true> Data Interfaces

| moneypunct_byname<wchar_t, true>::inti(GLIBCXX_3.4) [ISOcxx] |
| typeinfo for moneypunct_byname<wchar_t, true>(GLIBCXX_3.4) [CXXABI-1.86] |
| typeinfo name for moneypunct_byname<wchar_t, true>(GLIBCXX_3.4) [CXXABI-1.86] |

| vtable for moneypunct_byname<wchar_t, true>(GLIBCXX_3.4) [CXXABI-1.86] |
16 Libraries

LSB Core - Generic 5.0

16.1.129 Class money_base

16.1.129.1 Class data for money_base

The Run Time Type Information for the std::money_base class is described by Table 16-389.

Table 16-389 typeinfo for money_base

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for money_base</td>
</tr>
</tbody>
</table>

16.1.129.2 Interfaces for Class money_base

An LSB conforming implementation shall provide the generic methods for Class std::money_base specified in Table 16-390, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-390 libstdcxx - Class money_base Function Interfaces

| money_base::_S_construct_pattern(char, char, char)(GLIBCXX_3.4) [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::money_base specified in Table 16-391, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-391 libstdcxx - Class money_base Data Interfaces

| money_base::_S_default_pattern(GLIBCXX_3.4) [ISOCXX] |
| money_base::_S_atoms(GLIBCXX_3.4) [ISOCXX] |
| typeinfo for money_base(GLIBCXX_3.4) [CXXABI-1.86] |
| typeinfo name for money_base(GLIBCXX_3.4) [CXXABI-1.86] |

16.1.130 Class money_get<char, istreambuf_iterator<char, char_traits<char> > >

16.1.130.1 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 by Table 16-392.

Table 16-392 Primary vtable for money_get<char, istreambuf_iterator<char, char_traits<char> > > >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;::money_get()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;::money_get()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;::money_get()</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the std::money_get<char, std::istreambuf_iterator<char, std::char_traits<char> > > class is described by Table 16-393.

### Table 16-393: Typeinfo for `money_get<char, istreambuf_iterator<char, char_traits<char> > >`

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>Vtable for _cxxabi\1::_si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class std::money_get<char, std::istreambuf_iterator<char, std::char_traits<char> > > specified in Table 16-395, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-395  libstdcxx - Class money_get<char, istreambuf_iterator<char, char_traits<char> > > > Data Interfaces

| guard variable for money_get<char, istreambuf_iterator<char, char_traits<char> > > >::id(GLIBCXX_3.4) | [CXXABI-1.86] |
| money_get<char, istreambuf_iterator<char, char_traits<char> > > >::id(GLIBCXX_3.4) | [CXXABI-1.86] |
| typeinfo for money_get<char, istreambuf_iterator<char, char_traits<char> > > >::typeinfo(GLIBCXX_3.4) | [CXXABI-1.86] |
| typeinfo name for money_get<char, istreambuf_iterator<char, char_traits<char> > > >::typeinfo_name(GLIBCXX_3.4) | [CXXABI-1.86] |
| vtable for money_get<char, istreambuf_iterator<char, char_traits<char> > > >::vtable(GLIBCXX_3.4) | [CXXABI-1.86] |

### 16.1.131 Class money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > >

#### 16.1.131.1 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 by Table 16-396

| Base Offset | 0 |
| Virtual Base Offset | 0 |
| RTTI | typeinfo for money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > > |

© 2015 Linux Foundation
| vfunc[0]: | money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::~money_get() |
| vfunc[1]: | money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::~money_get() |
| vfunc[2]: | money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::do_get(istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, bool, ios_base&, _Ios_Iostate&, long double&) const |
| vfunc[3]: | money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::do_get(istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, bool, ios_base&, _Ios_Iostate&, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >&) const |

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 16-397.

Table 16-397 typeinfo for money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > class is described by Table 16-397.

| Base Vtable | vtable for __cxxabiv1::__si_class_type_info |
| Name | typeinfo name for money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > |

16.1.131.2 Interfaces for Class std::money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > >

An LSB conforming implementation shall provide the generic methods for Class std::money_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 16-398, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-398 libstdcxx - Class money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > > Function Interfaces

| istreambuf_iterator<wchar_t, char_traits<wchar_t> > > money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > |
| istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::M_extract<false>(istreambuf_iterator<wchar_t, char_traits<wchar_t> > >, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >, ios_base&, _Ios_Iostate&, basic_string<char, char_traits<char>, allocator<char> >& ) const(GLIBCXX_3.4) |
| istreambuf_iterator<wchar_t, char_traits<wchar_t> > > money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > |

© 2015 Linux Foundation
An LSB conforming implementation shall provide the generic data interfaces for Class std::money_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 16-399, with the full mandatory functionality as described in the referenced underlying specification.

### Table 16-399 libstdcxx - Class money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>guard variable for money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;=</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::id(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::id(GLIBCXX_3.4)</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
<tr>
<td>vtable for money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</td>
<td>GLIBCXX_3.4 [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

### 16.1.132 Class money_put<char, ostreambuf_iterator<char, char_traits<char> > >

#### 16.1.132.1 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, char_traits<char> > >
Table 16-400 Primary vtable for money_put<char, ostreambuf_iterator<char, char_traits<char>> >>

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt; &gt;&gt;</td>
</tr>
</tbody>
</table>

vfunc[0]: money_put<char, ostreambuf_iterator<char, char_traits<char>>> >> ::~money_put()

vfunc[1]: money_put<char, ostreambuf_iterator<char, char_traits<char>>> >> ::~money_put()

vfunc[2]: money_put<char, ostreambuf_iterator<char, char_traits<char>>> >> ::do_put(ostreambuf_iterator<char, char_traits<char>>, bool, ios_base&, char, long double) const

vfunc[3]: money_put<char, ostreambuf_iterator<char, char_traits<char>>> >> ::do_put(ostreambuf_iterator<char, char_traits<char>>, bool, ios_base&, char, basic_string<char, char_traits<char>, allocator<char>> const&) const

The Run Time Type Information for the std::money_put<char, ostreambuf_iterator<char, std::char_traits<char>>> class is described by Table 16-401.

Table 16-401 typeinfo for money_put<char, ostreambuf_iterator<char, char_traits<char>> >>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt; &gt;&gt;</td>
</tr>
</tbody>
</table>

16.1.132.2 Interfaces for Class money_put<char, ostreambuf_iterator<char, char_traits<char>>> >>

An LSB conforming implementation shall provide the generic methods for Class std::money_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>> >> specified in Table 16-402, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-402 libstdcxx - Class money_put<char, ostreambuf_iterator<char, char_traits<char>> >> Function Interfaces

| money_put<char, ostreambuf_iterator<char, char_traits<char>>> >> ::put(ostreambuf_iterator<char, char_traits<char>>, bool, ios_base&, char, basic_string<char, char_traits<char>, allocator<char>> const&) |
| money_put<char, ostreambuf_iterator<char, char_traits<char>>> >> |

© 2015 Linux Foundation
An LSB conforming implementation shall provide the generic data interfaces for Class
std::money_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > specified in
Table 16-403, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-403 libstdcxx - Class money_put<char, std::ostreambuf_iterator<char, char_traits<char> > > Data Interfaces

<table>
<thead>
<tr>
<th>Description</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>guard variable for money_put&lt;char, std::ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>GLIBCXX 3.4</td>
</tr>
<tr>
<td>money_put&lt;char, std::ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>GLIBCXX 3.4</td>
</tr>
<tr>
<td>typeinfo for money_put&lt;char, std::ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>GLIBCXX 3.4</td>
</tr>
<tr>
<td>vtable for money_put&lt;char, std::ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
<td>GLIBCXX 3.4</td>
</tr>
</tbody>
</table>

16.1.133 Class money_put<wchar_t,
std::ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

16.1.133.1 Class data for money_put<wchar_t,
std::ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

The virtual table for the std::money_put<wchar_t, std::ostreambuf_iterator<wchar_t, char_traits<wchar_t>> class is as follows:
std::char_traits<wchar_t> > > class is described by Table 16-404

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>vfunc[0]:</th>
<th>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::__money_put()</th>
</tr>
</thead>
<tbody>
<tr>
<td>vfunc[1]:</td>
<td>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::__money_put()</td>
</tr>
<tr>
<td>vfunc[2]:</td>
<td>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::do_put(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, bool, ios_base&amp;, wchar_t, long double) const</td>
</tr>
<tr>
<td>vfunc[3]:</td>
<td>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::do_put(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, bool, ios_base&amp;, wchar_t, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;) const</td>
</tr>
</tbody>
</table>

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 16-405

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</td>
</tr>
</tbody>
</table>

16.1.133.2 Interfaces for Class money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > >

An LSB conforming implementation shall provide the generic methods for Class std::money_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > > specified in Table 16-406, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;___put(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, bool, ios_base&amp;, wchar_t, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;) const(GLIBCXX_3.4) [ISO.CXX]</th>
<th>Function Interfaces</th>
</tr>
</thead>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class `std::money_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > >` specified in Table 16-407, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-407 libstdcxx - Class `money_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > >` Data Interfaces

<table>
<thead>
<tr>
<th>Guard variable for</th>
<th>money_put&lt;wchar_t, std::ostreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;</th>
<th>&gt;</th>
<th>id(GLIBCXX_3.4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typeinfo for</td>
<td>money_put&lt;wchar_t, std::ostreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;</td>
<td>&gt;</td>
<td>id(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Typeinfo name for</td>
<td>money_put&lt;wchar_t, std::ostreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;</td>
<td>&gt;</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Vtable for</td>
<td>money_put&lt;wchar_t, std::ostreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;</td>
<td>&gt;</td>
<td>(GLIBCXX_3.4)</td>
</tr>
</tbody>
</table>

16.1.134 Class locale

16.1.134.1 Interfaces for Class locale

An LSB conforming implementation shall provide the generic methods for Class `std::locale` specified in Table 16-408, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-408 libstdcxx - Class `locale` Data Interfaces

<table>
<thead>
<tr>
<th>Guard variable for</th>
<th>money_put&lt;wchar_t, std::ostreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;</th>
<th>&gt;</th>
<th>id(GLIBCXX_3.4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typeinfo for</td>
<td>money_put&lt;wchar_t, std::ostreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;</td>
<td>&gt;</td>
<td>id(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Typeinfo name for</td>
<td>money_put&lt;wchar_t, std::ostreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;</td>
<td>&gt;</td>
<td>(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Vtable for</td>
<td>money_put&lt;wchar_t, std::ostreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;</td>
<td>&gt;</td>
<td>(GLIBCXX_3.4)</td>
</tr>
</tbody>
</table>
An LSB conforming implementation shall provide the generic data interfaces for Class std::locale specified in Table 16-409, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-409 libstdcxx - Class locale Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>locale::all</td>
<td>Returns all facets</td>
</tr>
<tr>
<td>locale::none</td>
<td>Returns no facets</td>
</tr>
<tr>
<td>locale::time</td>
<td>Returns time facets</td>
</tr>
<tr>
<td>locale::ctype</td>
<td>Returns character classification</td>
</tr>
<tr>
<td>locale::collate</td>
<td>Returns collate facets</td>
</tr>
<tr>
<td>locale::numeric</td>
<td>Returns numeric facets</td>
</tr>
<tr>
<td>locale::messages</td>
<td>Returns message facets</td>
</tr>
<tr>
<td>locale::monetary</td>
<td>Returns monetary facets</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
### 16.1.135 Class locale::facet

#### 16.1.135.1 Class data for locale::facet

The virtual table for the std::locale::facet class is described by Table 16-410:

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for locale::facet</td>
</tr>
<tr>
<td>vfunc[0]:</td>
<td>locale::facet::~facet()</td>
</tr>
<tr>
<td>vfunc[1]:</td>
<td>locale::facet::~facet()</td>
</tr>
</tbody>
</table>

The Run Time Type Information for the std::locale::facet class is described by Table 16-411:

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for locale::facet</td>
</tr>
</tbody>
</table>

#### 16.1.135.2 Interfaces for Class locale::facet

An LSB conforming implementation shall provide the generic methods for Class std::locale::facet specified in Table 16-412, with the full mandatory functionality as described in the referenced underlying specification:

| locale::facet::_S_get_c_name() | GLIBCXX_3.4.6 | [ISOCXX] |
| locale::facet::_S_get_c_locale() | GLIBCXX_3.4 | [ISOCXX] |
| locale::facet::_S_clone_c_locale(__locale_struct*&) | GLIBCXX_3.4 | [ISOCXX] |
| locale::facet::_S_create_c_locale(__locale_struct*&, char const*, __locale_struct*) | GLIBCXX_3.4 | [ISOCXX] |
| locale::facet::_S_destroy_c_locale(__locale_struct*&) | GLIBCXX_3.4 | [ISOCXX] |
| locale::locale(locale const&, char const*, int) | GLIBCXX_3.4 | [ISOCXX] |

An LSB conforming implementation shall provide the generic data interfaces for Class std::locale::facet specified in Table 16-413, with the full mandatory functionality as described in the referenced underlying specification:

| __timepunct_cache<char>::_S_timezones(GLIBCXX_3.4) | GLIBCXX_3.4 | [ISOCXX] |
| __timepunct_cache<wchar_t>::_S_timezones(GLIBCXX_3.4) | GLIBCXX_3.4 | [ISOCXX] |
| typeinfo for locale::facet(GLIBCXX_3.4) | GLIBCXX_3.4 | [ISOCXX] |
| typeinfo name for locale::facet(GLIBCXX_3.4) | GLIBCXX_3.4 | [ISOCXX] |
| vtable for locale::facet(GLIBCXX_3.4) | GLIBCXX_3.4 | [ISOCXX] |
16.1.136 facet functions

16.1.136.1 Interfaces for facet functions

An LSB conforming implementation shall provide the generic methods for facet functions specified in Table 16-414, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function Description</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>void __convert_to_v&lt;double&gt;(char const*, double&amp;, __locale_struct* const&amp;)</td>
<td>(GLIBCXX_3.4) [ISOcXX]</td>
</tr>
<tr>
<td>void __convert_to_v&lt;long double&gt;(char const*, long double&amp;, __locale_struct* const&amp;)</td>
<td>(GLIBCXX_3.4) [ISOcXX]</td>
</tr>
<tr>
<td>void __convert_to_v&lt;float&gt;(char const*, float&amp;, __locale_struct* const&amp;)</td>
<td>(GLIBCXX_3.4) [ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;moneypunct&lt;char, false&gt; &gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;moneypunct&lt;wchar_t, false&gt; &gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;ctype&lt;wchar_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;codecvt&lt;char, char, __mbstate_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;codecvt&lt;wchar_t, char, __mbstate_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;collate&lt;char&gt; &gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;collate&lt;wchar_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;messages&lt;char&gt; &gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;messages&lt;wchar_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;numpunct&lt;char&gt; &gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;numpunct&lt;wchar_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
<tr>
<td>bool has_facet&lt;money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISOcXX]</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Standards</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool has_facet&lt;money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>bool has_facet&lt;money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>money_put&lt;char, false&gt; const&amp; use_facet&lt;money_put&lt;char, false&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>money_put&lt;wchar_t, false&gt; const&amp; use_facet&lt;money_put&lt;wchar_t, false&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>money_put&lt;wchar_t, true&gt; const&amp; use_facet&lt;money_put&lt;wchar_t, true&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>__timepunct&lt;char&gt; const&amp; use_facet&lt;__timepunct&lt;char&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>__timepunct&lt;wchar_t&gt; const&amp; use_facet&lt;__timepunct&lt;wchar_t&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>ctype&lt;char&gt; const&amp; use_facet&lt;ctype&lt;char&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>ctype&lt;wchar_t&gt; const&amp; use_facet&lt;ctype&lt;wchar_t&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>codecvt&lt;char, char, __mbstate_t&gt; const&amp; use_facet&lt;codecvt&lt;char, char, __mbstate_t&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>codecvt&lt;wchar_t, char, __mbstate_t&gt; const&amp; use_facet&lt;codecvt&lt;wchar_t, char, __mbstate_t&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>collate&lt;char&gt; const&amp; use_facet&lt;collate&lt;char&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>collate&lt;wchar_t&gt; const&amp; use_facet&lt;collate&lt;wchar_t&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; const&amp; use_facet&lt;num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; const&amp; use_facet&lt;num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; const&amp; use_facet&lt;num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; const&amp; use_facet&lt;num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>messages&lt;char&gt; const&amp; use_facet&lt;messages&lt;char&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>messages&lt;wchar_t&gt; const&amp; use_facet&lt;messages&lt;wchar_t&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>numpunct&lt;char&gt; const&amp; use_facet&lt;numpunct&lt;char&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>numpunct&lt;wchar_t&gt; const&amp; use_facet&lt;numpunct&lt;wchar_t&gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
<tr>
<td>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; const&amp; use_facet&lt;time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; &gt;&gt;(locale const&amp;)(GLIBCXX_3.4)</td>
<td>[ISO CXX]</td>
<td></td>
</tr>
</tbody>
</table>
16.1.137 Class __num_base

16.1.137.1 Class data for __num_base

16.1.137.2 Interfaces for Class __num_base

An LSB conforming implementation shall provide the generic methods for Class std::__num_base specified in Table 16-415, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-415 libstdcxx - Class __num_base Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>__num_base::_S_format_float(ios_base const&amp;, char*, char)</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

An LSB conforming implementation shall provide the generic data interfaces for Class std::__num_base specified in Table 16-416, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-416 libstdcxx - Class __num_base Data Interfaces

<table>
<thead>
<tr>
<th>Data Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>__num_base::_S_atoms_in()</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>__num_base::_S_atoms_out()</td>
<td>(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.138 Class num_get<char, istreambuf_iterator<char, char_traits<char>> > > >

16.1.138.1 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 by Table 16-417.
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base Offset</strong></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td><strong>Virtual Base Offset</strong></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td><strong>RTTI</strong></td>
<td>typeinfo for num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt; &gt;</td>
<td></td>
</tr>
<tr>
<td><strong>vfunc[0]:</strong></td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt; &gt;::~num_get()</td>
<td></td>
</tr>
<tr>
<td><strong>vfunc[1]:</strong></td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt; &gt;::~num_get()</td>
<td></td>
</tr>
<tr>
<td><strong>vfunc[2]:</strong></td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;::do_get(istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, bool&amp;) const</td>
<td></td>
</tr>
<tr>
<td><strong>vfunc[3]:</strong></td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;::do_get(istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, long&amp;) const</td>
<td></td>
</tr>
<tr>
<td><strong>vfunc[4]:</strong></td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;::do_get(istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, unsigned short&amp;) const</td>
<td></td>
</tr>
<tr>
<td><strong>vfunc[5]:</strong></td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;::do_get(istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, unsigned int&amp;) const</td>
<td></td>
</tr>
<tr>
<td><strong>vfunc[6]:</strong></td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;::do_get(istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, unsigned long&amp;) const</td>
<td></td>
</tr>
<tr>
<td><strong>vfunc[7]:</strong></td>
<td>num_get&lt;char,</td>
<td></td>
</tr>
</tbody>
</table>
The Run Time Type Information for the std::num_get<char, istreambuf_iterator<char, std::char_traits<char> > > class is described by Table 16-418.

<table>
<thead>
<tr>
<th>Vfunc Index</th>
<th>Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>vfunc[8]</td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; do_get(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, long long&amp;) const</td>
</tr>
<tr>
<td>vfunc[9]</td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; do_get(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, unsigned long long&amp;) const</td>
</tr>
<tr>
<td>vfunc[10]</td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; do_get(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, float&amp;) const</td>
</tr>
<tr>
<td>vfunc[11]</td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; do_get(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, double&amp;) const</td>
</tr>
<tr>
<td>vfunc[12]</td>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; do_get(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, void*) const</td>
</tr>
</tbody>
</table>

Table 16-418 typeinfo for num_get<char, istreambuf_iterator<char, char_traits<char> > > >
16.1.138.2 Interfaces for Class `num_get<char, istreambuf_iterator<char, char_traits<char> > >` > >

An LSB conforming implementation shall provide the generic methods for Class `std::num_get<char, std::istreambuf_iterator<char, std::char_traits<char> > >` > > specified in Table 16-419, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; ::M_extract_int&lt;unsigned int&gt;(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, unsigned int&amp;) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; ::M_extract_int&lt;long&gt;(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, long&amp;) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; ::M_extract_int&lt;unsigned long&gt;(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, unsigned long&amp;) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; ::M_extract_int&lt;unsigned short&gt;(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, unsigned short&amp;) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; ::M_extract_int&lt;long long&gt;(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, long long&amp;) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; ::M_extract_int&lt;long&gt;(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, long&amp;) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; ::M_extract_float(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&amp;) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; ::M_extract_float(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&amp;) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt; ::M_extract_float(istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&amp;) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

Table 16-419 libstdcxx - Class `num_get<char, istreambuf_iterator<char, char_traits<char> > > > Function Interfaces

<table>
<thead>
<tr>
<th>Type Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>__cxxabiv1::__si_class_type_info</td>
</tr>
<tr>
<td>Name: typeinfo name for num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</td>
</tr>
<tr>
<td>basetype: typeinfo for locale::facet</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, bool&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, double&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, long double&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, float&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, unsigned int&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, long&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, unsigned long&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, unsigned short&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, long long&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :do_get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, void*) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :do_get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, bool&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :do_get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, double&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
> :do_get(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, long double&) const(GLIBCXX_3.4) [ISOCXX]

num_get<char, istreambuf_iterator<char, char_traits<char> > >
An LSB conforming implementation shall provide the generic data interfaces for Class `std::num_get<char, std::istreambuf_iterator<char, std::char_traits<char>>>` specified in Table 16-420, with the full mandatory functionality as described in the referenced underlying specification.

**Table 16-420 libstdcxx - Class num_get<char, istreambuf_iterator<char, char_traits<char>> > > Guard Data Interfaces**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::id()</td>
<td>guard variable for num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;</td>
</tr>
<tr>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::id()</td>
<td>typeinfo for num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;</td>
</tr>
<tr>
<td>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::id()</td>
<td>typeinfo name for num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;</td>
</tr>
</tbody>
</table>
16.1.139 Class num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

16.1.139.1 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 by Table 16-421.

Table 16-421 Primary vtable for num_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual Base Offset</td>
<td>0</td>
</tr>
<tr>
<td>RTTI</td>
<td>typeinfo for num_get&lt;wchar_t, std::istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>vfunk[0]:</th>
<th>num_get&lt;wchar_t, std::istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;::~num_get()</th>
</tr>
</thead>
<tbody>
<tr>
<td>vfunk[1]:</td>
<td>num_get&lt;wchar_t, std::istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;::~num_get()</td>
</tr>
<tr>
<td>vfunk[2]:</td>
<td>num_get&lt;wchar_t, std::istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;::do_get(istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt;, std::ios_base&amp;, _Ios_Iostate&amp;, bool&amp;) const</td>
</tr>
<tr>
<td>vfunk[3]:</td>
<td>num_get&lt;wchar_t, std::istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;::do_get(istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt;, std::ios_base&amp;, _Ios_Iostate&amp;, long&amp;) const</td>
</tr>
<tr>
<td>vfunk[4]:</td>
<td>num_get&lt;wchar_t, std::istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;::do_get(istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt;, std::ios_base&amp;, _Ios_Iostate&amp;, unsigned short&amp;) const</td>
</tr>
<tr>
<td>vfunk[5]:</td>
<td>num_get&lt;wchar_t, std::istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt; &gt;::do_get(istreambuf_iterator&lt;wchar_t, std::char_traits&lt;wchar_t&gt; &gt;, std::ios_base&amp;, _Ios_Iostate&amp;, long&amp;) const</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
| vfunc[6]: | num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>, char_traits<wchar_t>, ios_base&), _Ios_Iostate&, unsigned int&) const |
| vfunc[7]: | num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>, char_traits<wchar_t>, ios_base&), _Ios_Iostate&, unsigned long&) const |
| vfunc[8]: | num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>, char_traits<wchar_t>, ios_base&), _Ios_Iostate&, long long&) const |
| vfunc[9]: | num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>, char_traits<wchar_t>, ios_base&), _Ios_Iostate&, float&) const |
| vfunc[10]: | num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>, char_traits<wchar_t>, ios_base&), _Ios_Iostate&, double&) const |
| vfunc[11]: | num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>, char_traits<wchar_t>, ios_base&), _Ios_Iostate&, long double&) const |
| vfunc[12]: | num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>, char_traits<wchar_t>> |

**16 Libraries**

**LSB Core - Generic 5.0**

© 2015 Linux Foundation
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 16-422.

Table 16-422 typeinfo for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > >>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;&gt;</td>
</tr>
<tr>
<td>basetype:</td>
<td>typeinfo for locale::facet</td>
</tr>
</tbody>
</table>

16.1.139.2 Interfaces for Class num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > >>

An LSB conforming implementation shall provide the generic methods for Class std::num_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > >> specified in Table 16-423, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-423 libstdcxx - Class num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > >> Function Interfaces

<table>
<thead>
<tr>
<th>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; ::M_extract_int&lt;unsigned int&gt;(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;), const(GLIBCXX_3.4) [ISOCXX]</th>
</tr>
</thead>
<tbody>
<tr>
<td>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; ::M_extract_int&lt;long&gt;(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;), const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; ::M_extract_int&lt;unsigned long&gt;(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;), const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; ::M_extract_int&lt;unsigned short&gt;(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;), const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; ::M_extract_int&lt;long long&gt;(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;), const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;&gt; ::M_extract_int&lt;unsigned long long&gt;(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;), const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>
# Libraries

**LSB Core - Generic 5.0**

- `istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, unsigned long long&) const(GLIBCXX_3.4)` [ISOCXX]

- `num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, basic_string<char, char_traits<char>, allocator<char>>&) const(GLIBCXX_3.4)` [ISOCXX]

- `num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, void*&) const(GLIBCXX_3.4)` [ISOCXX]

- `num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, bool&) const(GLIBCXX_3.4)` [ISOCXX]

- `num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, long double&) const(GLIBCXX_3.4)` [ISOCXX]

- `num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, unsigned int&) const(GLIBCXX_3.4)` [ISOCXX]

- `num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, long&) const(GLIBCXX_3.4)` [ISOCXX]

- `num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, unsigned long&) const(GLIBCXX_3.4)` [ISOCXX]

- `num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, unsigned short&) const(GLIBCXX_3.4)` [ISOCXX]

- `num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, unsigned long long&) const(GLIBCXX_3.4)` [ISOCXX]

- `num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, ios_base&, _Ios_Iostate&, unsigned short&) const(GLIBCXX_3.4)` [ISOCXX]
An LSB conforming implementation shall provide the generic data interfaces for Class
std::num_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > >
specified in Table 16-424, with the full mandatory functionality as described in the referenced underlying specification.
### 16.1.140 Class num_put<char, ostreambuf_iterator<char, char_traits<char>> > >

#### 16.1.140.1 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 by Table 16-425

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]:</th>
<th>vfunc[1]:</th>
<th>vfunc[2]:</th>
<th>vfunc[3]:</th>
<th>vfunc[4]:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td>num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt; ::~num_put()</td>
<td>num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt; ::~num_put()</td>
<td>num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt; ::do_put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;, ios_base&amp;, char, bool) const</td>
<td>num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt; ::do_put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;, ios_base&amp;, char, long) const</td>
<td>num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt; ::do_put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;, ios_base&amp;, char, unsigned long) const</td>
</tr>
</tbody>
</table>
The Run Time Type Information for the `std::num_put<char, ostreambuf_iterator<char, char_traits<char> > >` class is described by Table 16-426.

Table 16-426: `typedef info for num_put<char, ostreambuf_iterator<char, char_traits<char> > >` >>

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;</td>
</tr>
<tr>
<td>basetype:</td>
<td>typeinfo for locale::facet</td>
</tr>
</tbody>
</table>

16.1.140.2 Interfaces for Class `std::num_put<char, ostreambuf_iterator<char, char_traits<char> > >` >>

An LSB conforming implementation shall provide the generic methods for Class `std::num_put<char, ostreambuf_iterator<char, char_traits<char> > >` >> specified in Table 16-427, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-427: libstdcxx - Class `num_put<char, ostreambuf_iterator<char, char_traits<char> > >` >> Function Interfaces

- `ostreambuf_iterator<char, char_traits<char> > >> num_put<char, ostreambuf_iterator<char, char_traits<char> > >`
16 Libraries

LSB Core - Generic 5.0

```cpp
>::_M_insert_int<long>(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, long) const(GLIBCXX_3.4) [ISOCXX]

ostreambuf_iterator<char, char_traits<char> > > num_put<char,
ostreambuf_iterator<char, char_traits<char> > > >:::_M_insert_int<unsigned long>(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, unsigned long) const(GLIBCXX_3.4) [ISOCXX]

ostreambuf_iterator<char, char_traits<char> > > num_put<char,
ostreambuf_iterator<char, char_traits<char> > > >:::_M_insert_int<long>(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, long long) const(GLIBCXX_3.4) [ISOCXX]

ostreambuf_iterator<char, char_traits<char> > > num_put<char,
ostreambuf_iterator<char, char_traits<char> > > >:::_M_insert_int<unsigned long long>(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, unsigned long long) const(GLIBCXX_3.4) [ISOCXX]

ostreambuf_iterator<char, char_traits<char> > > num_put<char,
ostreambuf_iterator<char, char_traits<char> > > >:::_M_insert_float<double>(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, char, double) const(GLIBCXX_3.4) [ISOCXX]

ostreambuf_iterator<char, char_traits<char> > > num_put<char,
ostreambuf_iterator<char, char_traits<char> > > >:::_M_insert_float<long double>(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, char, long double) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, void const*) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, bool) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, double) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, long double) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, long) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, unsigned long) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, unsigned long long) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::do_put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, void const*) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::do_put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, bool) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::do_put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, double) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::do_put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, long double) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::do_put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, long) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::do_put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, unsigned long) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::do_put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, unsigned long long) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::do_put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, void const*) const(GLIBCXX_3.4) [ISOCXX]

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::do_put(ostreambuf_iterator<char, char_traits<char> >, ios_base&, char, bool) const(GLIBCXX_3.4) [ISOCXX]
```

© 2015 Linux Foundation
An LSB conforming implementation shall provide the generic data interfaces for Class std::num_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > specified in Table 16-428, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-428 libstdcxx - Class num_put<char, ostreambuf_iterator<char, char_traits<char> > > Data Interfaces

16.1.141 Class num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

16.1.141.1 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 by Table 16-429.

© 2015 Linux Foundation
### Table 16-429 Primary vtable for num_put<wchar_t, ostreambuf_iterator<wchar_t>, char_traits<wchar_t> >

<table>
<thead>
<tr>
<th>Base Offset</th>
<th>Virtual Base Offset</th>
<th>RTTI</th>
<th>vfunc[0]:</th>
<th>vfunc[1]:</th>
<th>vfunc[2]:</th>
<th>vfunc[3]:</th>
<th>vfunc[4]:</th>
<th>vfunc[5]:</th>
<th>vfunc[6]:</th>
<th>vfunc[7]:</th>
<th>vfunc[8]:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t&gt;, char_traits&lt;wchar_t&gt; &gt; ::~num_put()</td>
<td>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t&gt;, char_traits&lt;wchar_t&gt; &gt; ::~num_put()</td>
<td>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t&gt;, char_traits&lt;wchar_t&gt; &gt; ::do_put(ostreambuf_iterator&lt;wchar_t&gt;, ios_base&amp;, wchar_t bool) const</td>
<td>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t&gt;, char_traits&lt;wchar_t&gt; &gt; ::do_put(ostreambuf_iterator&lt;wchar_t&gt;, ios_base&amp;, wchar_t long) const</td>
<td>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t&gt;, char_traits&lt;wchar_t&gt; &gt; ::do_put(ostreambuf_iterator&lt;wchar_t&gt;, ios_base&amp;, wchar_t unsigned long) const</td>
<td>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t&gt;, char_traits&lt;wchar_t&gt; &gt; ::do_put(ostreambuf_iterator&lt;wchar_t&gt;, ios_base&amp;, wchar_t long long) const</td>
<td>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t&gt;, char_traits&lt;wchar_t&gt; &gt; ::do_put(ostreambuf_iterator&lt;wchar_t&gt;, ios_base&amp;, wchar_t double) const</td>
<td>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t&gt;, char_traits&lt;wchar_t&gt; &gt; ::do_put(ostreambuf_iterator&lt;wchar_t&gt;, ios_base&amp;, wchar_t long long) const</td>
<td></td>
</tr>
</tbody>
</table>
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 16-430.

Table 16-430 typeinfo for num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > class

<table>
<thead>
<tr>
<th>Base Vtable</th>
<th>vtable for __cxxabiv1::__si_class_type_info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>typeinfo name for num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</td>
</tr>
<tr>
<td>basetype</td>
<td>typeinfo for locale::facet</td>
</tr>
</tbody>
</table>

16.1.141.2 Interfaces for Class num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the generic methods for Class std::num_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > > with the full mandatory functionality as described in the referenced underlying specification.

Table 16-431 libstdcxx - Class num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > Function Interfaces

<table>
<thead>
<tr>
<th>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; _M_insert_int&lt;long&gt;(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, wchar_t, long) const(GLIBCXX_3.4) [ISOCXX]</th>
</tr>
</thead>
<tbody>
<tr>
<td>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; _M_insert_int&lt;unsigned long&gt;(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, wchar_t, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; _M_insert_int&lt;long double&gt;(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, wchar_t, long double) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; _M_insert_float&lt;double&gt;(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, wchar_t, char, double) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt; _M_insert_float&lt;long double&gt;(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, wchar_t, char, long double) const(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
An LSB conforming implementation shall provide the generic data interfaces for Class std::num_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 16-432, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-432: libstdcxx - Class num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > Data Interfaces

<table>
<thead>
<tr>
<th>Guard variable for num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::id(GLIBCXX_3.4) [CXXABI-1.86]</th>
</tr>
</thead>
<tbody>
<tr>
<td>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::id(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>Typeinfo for num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::id(GLIBCXX_3.4)</td>
</tr>
<tr>
<td>Typeinfo name for num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::id(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
<tr>
<td>Vtable for num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::id(GLIBCXX_3.4) [CXXABI-1.86]</td>
</tr>
</tbody>
</table>

16.1.142 Class __basic_file<char>

16.1.142.1 Class data for __basic_file<char>

16.1.142.2 Interfaces for Class __basic_file<char>

An LSB conforming implementation shall provide the generic methods for Class std::__basic_file<char> specified in Table 16-433, with the full mandatory functionality as described in the referenced underlying specification.

Table 16-433: libstdcxx - Class __basic_file<char> Function Interfaces

<table>
<thead>
<tr>
<th>__basic_file&lt;char&gt;::is_open() const(GLIBCXX_3.4) [ISOCXX]</th>
</tr>
</thead>
<tbody>
<tr>
<td>__basic_file&lt;char&gt;::fd() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>__basic_file&lt;char&gt;::file(int) (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>__basic_file&lt;char&gt;::open(char const*, _Ios_Openmode, int) (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>__basic_file&lt;char&gt;::sync() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>__basic_file&lt;char&gt;::close() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>__basic_file&lt;char&gt;::sys_open(_IO_FILE*, _Ios_Openmode) (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>__basic_file&lt;char&gt;::sys_open(int, _Ios_Openmode) (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>__basic_file&lt;char&gt;::showmanyc() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>__basic_file&lt;char&gt;:::__basic_file(pthread_mutex_t*) (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>__basic_file&lt;char&gt;:::~__basic_file() (GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.143 Class _List_node_base

16.1.143.1 Interfaces for Class _List_node_base

An LSB conforming implementation shall provide the generic methods for Class std::_List_node_base specified in Table 16-434, with the full mandatory functionality as
described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 16-434 libstdcxx - Class _List_node_base Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>_List_node_base::hook(_List_node_base*)(GLIBCXX_3.4) [LSB]</td>
</tr>
<tr>
<td>_List_node_base::swap(_List_node_base&amp;, _List_node_base&amp;)(GLIBCXX_3.4) [LSB]</td>
</tr>
<tr>
<td>_List_node_base::unhook()(GLIBCXX_3.4) [LSB]</td>
</tr>
<tr>
<td>_List_node_base::reverse()(GLIBCXX_3.4) [LSB]</td>
</tr>
<tr>
<td>_List_node_base::transfer(_List_node_base*, _List_node_base*)(GLIBCXX_3.4) [LSB]</td>
</tr>
</tbody>
</table>

16.1.144 Class allocator<char>

16.1.144.1 Class data for allocator<char>

16.1.144.2 Interfaces for Class allocator<char>

An LSB conforming implementation shall provide the generic methods for Class std::allocator<char> specified in Table 16-435, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 16-435 libstdcxx - Class allocator&lt;char&gt; Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>allocator&lt;char&gt;::allocator(allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>allocator&lt;char&gt;::allocator()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>allocator&lt;char&gt;::allocator(allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>allocator&lt;char&gt;::allocator()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>allocator&lt;char&gt;::~allocator()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>allocator&lt;char&gt;::~allocator()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.145 Class allocator<wchar_t>

16.1.145.1 Class data for allocator<wchar_t>

16.1.145.2 Interfaces for Class allocator<wchar_t>

An LSB conforming implementation shall provide the generic methods for Class std::allocator<wchar_t> specified in Table 16-436, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Table 16-436 libstdcxx - Class allocator&lt;wchar_t&gt; Function Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>allocator&lt;wchar_t&gt;::allocator(allocator&lt;wchar_t&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>allocator&lt;wchar_t&gt;::allocator()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>allocator&lt;wchar_t&gt;::allocator(allocator&lt;wchar_t&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>allocator&lt;wchar_t&gt;::allocator()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>allocator&lt;wchar_t&gt;::~allocator()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
<tr>
<td>allocator&lt;wchar_t&gt;::~allocator()(GLIBCXX_3.4) [ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.146 Class __gnu_cxx::__pool>true<

16.1.146.1 Interfaces for Class __gnu_cxx::__pool>true<

An LSB conforming implementation shall provide the generic methods for Class
__gnu_cxx::__pool<true> specified in Table 16-437, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Version</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>__gnu_cxx::__pool&lt;true&gt;::_M_destroy()</td>
<td>GLIBCXX_3.4.4</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__gnu_cxx::__pool&lt;true&gt;::_M_init()</td>
<td>GLIBCXX_3.4.6</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__gnu_cxx::__pool&lt;true&gt;::_M_get_thread_id()</td>
<td>GLIBCXX_3.4.4</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__gnu_cxx::__pool&lt;true&gt;::_M_destroy_thread_key()</td>
<td>GLIBCXX_3.4.4</td>
<td>[LSB]</td>
</tr>
</tbody>
</table>

16.1.147 Class __gnu_cxx::__pool<false>

16.1.147.1 Interfaces for Class __gnu_cxx::__pool<false>

An LSB conforming implementation shall provide the generic methods for Class __gnu_cxx::__pool<false> specified in Table 16-438, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Version</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>__gnu_cxx::__pool&lt;false&gt;::_M_destroy()</td>
<td>GLIBCXX_3.4.4</td>
<td>[LSB]</td>
</tr>
<tr>
<td>__gnu_cxx::__pool&lt;false&gt;::_M_init()</td>
<td>GLIBCXX_3.4.4</td>
<td>[LSB]</td>
</tr>
</tbody>
</table>

16.1.148 Class __gnu_cxx::free_list

16.1.148.1 Interfaces for Class __gnu_cxx::free_list

An LSB conforming implementation shall provide the generic methods for Class __gnu_cxx::free_list specified in Table 16-439, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Version</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>__gnu_cxx::free_list::_M_clear()</td>
<td>GLIBCXX_3.4.4</td>
<td>[LSB]</td>
</tr>
</tbody>
</table>

16.1.149 Class char_traits<char>

16.1.149.1 Interfaces for Class char_traits<char>

An LSB conforming implementation shall provide the generic methods for Class std::char_traits<char> specified in Table 16-440, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Version</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>char_traits&lt;char&gt;::eq(char const&amp;, char const&amp;)</td>
<td>GLIBCXX_3.4.5</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>

16.1.150 Class char_traits<wchar_t>

16.1.150.1 Interfaces for Class char_traits<wchar_t>

An LSB conforming implementation shall provide the generic methods for Class std::char_traits<wchar_t> specified in Table 16-441, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Function</th>
<th>Version</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>char_traits&lt;wchar_t&gt;::eq(wchar_t const&amp;, wchar_t const&amp;)</td>
<td>GLIBCXX_3.4.5</td>
<td>[ISOCXX]</td>
</tr>
</tbody>
</table>
16.2 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 16.1 shall behave as described in the referenced base document.
17 Commands and Utilities

17.1 Commands and Utilities

An LSB conforming implementation shall provide the commands and utilities as described in Table 17-1, with at least the behavior described as mandatory in the referenced underlying specification, with the following exceptions:

1. If any operand (except one which follows --) starts with a hyphen, the behavior is unspecified.

Rationale (Informative): Applications should place options before operands, or use -- as needed. This text is needed because, by default, GNU option parsing differs from POSIX, unless the environment variable POSIXLY_CORRECT is set. For example, ls . -a in GNU ls means to list the current directory, showing all files (that is, "." is an operand and -a is an option). In POSIX, "." and -a are both operands, and the command means to list the current directory, and also the file named -a. Suggesting that applications rely on the setting of the POSIXLY_CORRECT environment variable, or try to set it, seems worse than just asking the applications to invoke commands in ways which work with either the POSIX or GNU behaviors.

Table 17-1 Commands And Utilities

|-----|--------|-------------|-------|------------|
# 17 Commands and Utilities

<table>
<thead>
<tr>
<th>diff</th>
<th>iconv</th>
<th>more</th>
<th>sleep</th>
<th>xargs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dirn</td>
<td>id</td>
<td>mount</td>
<td>sort</td>
<td>zcat</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dmesg</td>
<td>info</td>
<td>msgfmt</td>
<td>split</td>
<td></td>
</tr>
<tr>
<td></td>
<td>cmp</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Referenced Specification(s)


[2]. This Specification

[3]. Libncursesw API

An LSB conforming implementation shall provide the shell built in utilities as described in Table 17-2, with at least the behavior described as mandatory in the referenced underlying specification, with the following exceptions:

1. The built in commands and utilities shall be provided by the `sh` utility itself, and need not be implemented in a manner so that they can be accessed via the exec family of functions as defined in POSIX 1003.1-2008 (ISO/IEC 9945-2009) and should not be invoked directly by those standard utilities that execute other utilities (env, find, nice, nohup, time, xargs).

Rationale (Informative): Since the built in utilities must affect the environment of the calling process, they have no effect when executed as a file.

Table 17-2 Built In Utilities

<table>
<thead>
<tr>
<th>alias</th>
<th>command</th>
<th>getopts</th>
<th>read</th>
<th>umask</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bg</td>
<td>fc</td>
<td>hash</td>
<td>type</td>
<td>unalias</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cd</td>
<td>fg</td>
<td>jobs</td>
<td>ulimit</td>
<td>wait</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Referenced Specification(s)


## 17.2 Command Behavior

This section contains descriptions for commands and utilities whose specified behavior in the LSB contradicts or extends the standards referenced. It also contains commands and utilities only required by the LSB and not specified by other standards.
ar

Name
ar — create and maintain library archives (DEPRECATED)

Description
ar is deprecated from the LSB and is expected to disappear from a future version of the LSB.

Rationale: The LSB generally does not include software development utilities nor does it specify .o and .a file formats.

ar is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009) but with differences as listed below.

Differences

-T
-C

need not be accepted.

-l

has unspecified behavior.

-q

has unspecified behavior; using -r is suggested.
at

Name
at — examine or delete jobs for later execution

Description
at is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009) but with differences as listed below.

Differences
Options
-d
is functionally equivalent to the -r option specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009).

-r
need not be supported, but the `-d' option is equivalent.

-t time
need not be supported.

Optional Control Files
The implementation shall support the XSI optional behavior for access control; however the files at.allow and at.deny may reside in /etc rather than /usr/lib/cron.

awk

Name
awk — pattern scanning and processing language

Description
awk is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009) but with differences as listed below.

Differences
Certain aspects of internationalized regular expressions are optional; see Regular Expressions.
batch

Name
batch — schedule commands to be executed in a batch queue

Description
The specification for batch is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

Optional Control Files
The implementation shall support the XSI optional behavior for access control; however the files at.allow and at.deny may reside in /etc rather than /usr/lib/cron.

bc

Name
bc — an arbitrary precision calculator language

Description
bc is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009) but with extensions as listed below.

Extensions
The bc language may be extended in an implementation defined manner. If an implementation supports extensions, it shall also support the additional options:

-s|--standard
   processes exactly the POSIX bc language.

-w|--warn
   gives warnings for extensions to POSIX bc.

chfn

Name
chfn — change user name and information

Synopsis
chfn [-f full_name] [-h home_phone] [user]

Description
chfn shall update the user database. An unprivileged user may only change the fields for their own account, a user with appropriate privileges may change the fields for any account.

The fields full_name and home_phone may contain any character except:
any control character
comma
colon
equal sign

If none of the options are selected, `chfn` operates in an interactive fashion. The prompts and expected input in interactive mode are unspecified and should not be relied upon.

As it is possible for the system to be configured to restrict which fields a non-privileged user is permitted to change, applications should be written to gracefully handle these situations.

**Standard Options**

- `f full_name`
  sets the user's full name.

- `h home_phone`
  sets the user's home phone number.

**Future Directions**

The following two options are expected to be added in a future version of the LSB:

- `o office`
  sets the user's office room number.

- `p office_phone`
  sets the user's office phone number.

Note that some implementations contain a "-o other" option which specifies an additional field called "other". Traditionally, this field is not subject to the constraints about legitimate characters in fields. Also, one traditionally shall have appropriate privileges to change the other field. At this point there is no consensus about whether it is desirable to specify the other field; applications may wish to avoid using it.

The "-w work_phone" field found in some implementations should be replaced by the "-p office_phone" field. The "-r room_number" field found in some implementations is the equivalent of the "-o office" option mentioned above; which one of these two options to specify will depend on implementation experience and the decision regarding the other field.
chsh

Name
chsh — change login shell

Synopsis
chsh [-s login_shell] [user]

Description
chsh changes the user login shell. This determines the name of the user's initial login command. An unprivileged user may only change the login shell for their own account, a user with appropriate privilege may change the login shell for any account specified by user.

Unless the user has appropriate privilege, the initial login command name shall be one of those listed in /etc/shells. The login_shell shall be the absolute path (i.e. it must start with '/') to an executable file. Accounts which are restricted (in an implementation-defined manner) may not change their login shell.

If the -s option is not selected, chsh operates in an interactive mode. The prompts and expected input in this mode are unspecified.

Standard Options

-s login_shell
sets the login shell.

col

Name
col — filter reverse line feeds from input

Description
col is as specified in SUSv2 but with differences as listed below.

Differences
The -p option has unspecified behavior.

Note: Although col is shown as legacy in SUSv2, it is not (yet) deprecated in the LSB.

cpio

Name
cpio — copy file archives in and out

Description
cpio is as specified in SUSv2, but with differences as listed below.

Differences
Some elements of the Pattern Matching Notation are optional; see Pattern Matching Notation.
crontab

Name
crontab — maintain crontab files for individual users

Synopsis
crontab [-u user] file crontab [-u user] {-l | -r | -e}

Description
crontab is as specified in POSIX.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

Optional Control Files
The implementation shall support the XSI optional behavior for access control; however the files cron.allow and cron.deny may reside in /etc rather than /usr/lib/cron.

df

Name
df — report file system disk space usage

Description
The df command shall behave as specified in POSIX.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

Differences

Options
If the -k option is not specified, disk space is shown in unspecified units. If the -P option is specified, the size of the unit shall be printed on the header line in the format "%s-blocks". Applications should specify -k.

The XSI option -t has unspecified behavior. Applications should not specify -t.

Rationale: The most common implementation of df uses the -t option for a different purpose (restricting output to a particular file system type), and use of -t is therefore non-portable.

Operand May Identify Special File
If an argument is the absolute file name of a special file containing a mounted file system, df shall show the space available on that file system rather than on the file system containing the special file (which is typically the root file system).

Note: In POSIX.1-2008 (ISO/IEC 9945-2009) the XSI optional behavior permits an operand to name a special file, but appears to require the operation be performed on the file system containing the special file. A defect report has been submitted for this case.
dmesg

Name
dmesg — print or control the system message buffer

Synopsis
dmesg [-c | -n level | -s bufsize]

Description
dmesg examines or controls the system message buffer. Only a user with appropriate privileges may modify the system message buffer parameters or contents.

Standard Options
-c
If the user has appropriate privilege, clears the system message buffer contents after printing.

-n level
If the user has appropriate privilege, sets the level at which logging of messages is done to the console.

-s bufsize
uses a buffer of bufsize to query the system message buffer. This is 16392 by default.

du

Name
du — estimate file space usage

Description
du is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

Differences
If the -k option is not specified, disk space is shown in unspecified units. Applications should specify -k.
echo

Name
  echo — write arguments to standard output

Synopsis
  echo [string...]

Description
  The echo command is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but
  with the following differences.

  Implementations may support implementation-defined options to echo. The behavior of
  echo if any arguments contain backslashes is also implementation defined.

Application Usage
  Conforming applications should not run echo with a first argument starting with a hy-
  phen, or with any arguments containing backslashes; they should use printf in those
  cases.

  Note: The behavior specified here is similar to that specified by POSIX 1003.1-2008
  (ISO/IEC 9945-2009) without the XSI option. However, the LSB strongly recommends con-
  forming applications not use any options (even if the implementation provides them) while
  POSIX 1003.1-2008 (ISO/IEC 9945-2009) specifies behavior if the first operand is the
  string -n.

egrep

Name
  egrep — search a file with an Extended Regular Expression pattern

Description
  egrep is equivalent to grep -E. For further details, see the specification for grep.

fgrep

Name
  fgrep — search a file with a fixed pattern

Description
  fgrep is equivalent to grep -F. For further details, see the specification for grep.
file

**Name**

`file` — determine file type

**Description**


**Differences**

The `-M`, `-h`, `-d`, and `-i` options need not be supported.

fuser

**Name**

`fuser` — identify processes using files or sockets

**Description**

`fuser` is as specified in [POSIX 1003.1-2008 (ISO/IEC 9945-2009)](http://pubs.opengroup.org/onlinepubs/9699919799/utilities/fuser.html), but with differences as listed below.

**Differences**

The `fuser` command is a system administration utility, see [Path For System Administration Utilities](http://www.linuxfoundation.org/collaborate/workgroups/server/utilities).

**Option Differences**

- `-c`
  
  has unspecified behavior.

- `-f`
  
  has unspecified behavior.
gettext

**Name**

gettext — retrieve text string from message catalog

**Synopsis**

`gettext` [options] [textdomain] msgid `gettext` -s [options] msgid...

**Description**

The `gettext` utility retrieves a translated text string corresponding to string `msgid` from a message object generated with `msgfmt` utility.

The message object name is derived from the optional argument `textdomain` if present, otherwise from the `TEXTDOMAIN` environment variable. If no domain is specified, or if a corresponding string cannot be found, `gettext` prints `msgid`.

Ordinarily `gettext` looks for its message object in `dirname/lang/LC_MESSAGES` where `dirname` is the implementation-defined default directory and `lang` is the locale name. If present, the `TEXTDOMAINDIR` environment variable replaces the `dirname`.

This utility interprets C escape sequences such as `\t` for tab. Use `\` to print a backslash. To produce a message on a line of its own, either put a `\n` at the end of `msgid`, or use this command in conjunction with the `printf` utility.

When used with the `-s` option the `gettext` utility behaves like the `echo` utility, except that the message corresponding to `msgid` in the selected catalog provides the arguments.

**Options**

- `-d domainname`
- `--domain=domainname`

PARAMETER translated messages from domainname.

- `-e`

Enable expansion of some escape sequences.

- `-n`

Suppress trailing newline.

**Operands**

The following operands are supported:

`textdomain`

A domain name used to retrieve the messages.

`msgid`

A key to retrieve the localized message.

**Environment Variables**

`LANGUAGE`

Specifies one or more locale names.

`LANG`
LSB Core - Generic 5.0

Specifies locale name.

**LC_MESSAGES**

Specifies messaging locale, and if present overrides LANG for messages.

**TEXTDOMAIN**

Specifies the text domain name, which is identical to the message object filename without .mo suffix.

**TEXTDOMAINDIR**

Specifies the pathname to the message catalog, and if present replaces the implementation-defined default directory.

**Exit Status**

The following exit values are returned:

0

Successful completion.

>0

An error occurred.

grep

**Name**

grep — print lines matching a pattern

**Description**

grep is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

**LSB Differences**

Certain aspects of regular expression matching are optional; see Regular Expressions.
groupadd

Name

groupadd — create a new group

Synopsis

groupadd [-g gid [-o]] group

Description

If the caller has appropriate privilege, the groupadd command shall create a new group named group. The group name shall be unique in the group database. If no gid is specified, groupadd shall create the new group with a unique group ID. If the group named group already exists, or if a group with gid ID exists and -o option is not set, groupadd shall issue a diagnostic message and exit with a non-zero exit status.

The groupadd command is a system administration utility, see Path For System Administration Utilities.

Options

-g gid [-o]

The new group shall have group ID gid. If the -o option is not used, no other group shall have this group ID. The value of gid shall be non-negative.

groupdel

Name

groupdel — delete a group

Synopsis

groupdel group

Description

If the caller has sufficient privilege, the groupdel command shall modify the system group database, deleting the group named group. If the group named group does not exist or is a primary group for existing user, groupdel shall issue a diagnostic message and exit with a non-zero exit status.

The groupdel command is a system administration utility, see Path For System Administration Utilities.
groupmod

Name

`groupmod` — modify a group

Synopsis

`groupmod [-g gid [-o]] [-n group_name] group`

Description

If the caller has appropriate privilege, the `groupmod` command shall modify the entry in the system group database corresponding to a group named `group`.

The `groupmod` command is a system administration utility, see [Path For System Administration Utilities](#).

Options

`-g gid [-o]`

Modify the group's group ID, setting it to `gid`. If the `-o` option is not used, no other group shall have this group ID. The value of `gid` shall be non-negative.

**Note:** Only the group ID in the database is altered; any files with group ownership set to the original group ID are unchanged by this modification.

`-n group_name`

changes the name of the group from `group` to `group_name`.

groups

Name

`groups` — display a group

Synopsis

`groups [user]`

Description

The `groups` command shall behave as `id -Gn [user]`, as specified in [POSIX 1003.1-2008 (ISO/IEC 9945-2009)](#). The optional `user` parameter will display the groups for the named user.

gunzip

Name

`gunzip` — uncompress files

Description

`gunzip` is equivalent to `gzip -d`. See the specification for `gzip` for further details.

[Filesystem Hierarchy Standard](#) requires that if `gunzip` exists, it must be a symbolic or hard link to `/bin/gzip`. This specification additionally allows `gunzip` to be a wrapper script which calls `gzip -d`. 
gzip

Name
gzip — compress or expand files

Synopsis
gzip [-cdfhllNrtvV19] [-S suffix] [name...]

Description
The gzip command shall attempt to reduce the size of the named files. Whenever possible, each file is replaced by one with the extension .gz, while keeping the same ownership, modes, access and modification times. If no files are specified, or if a file name is -, the standard input is compressed to the standard output. gzip shall only attempt to compress regular files. In particular, it will ignore symbolic links.


Options
-c, --stdout, --to-stdout
writes output on standard output, leaving the original files unchanged. If there are several input files, the output consists of a sequence of independently compressed members. To obtain better compression, concatenate all input files before compressing them.

-d, --decompress, --uncompress
the name operands are compressed files, and gzip shall decompress them.

-f, --force
forces compression or decompression even if the file has multiple links or the corresponding file already exists, or if the compressed data is read from or written to a terminal. If the input data is not in a format recognized by gzip, and if the option --stdout is also given, copy the input data without change to the standard output: let gzip behave as cat. If -f is not given, and when not running in the background, gzip prompts to verify whether an existing file should be overwritten.

-l, --list
lists the compressed size, uncompressed size, ratio and uncompressed name for each compressed file. For files that are not in gzip format, the uncompressed size shall be given as -1. If the --verbose or -v option is also specified, the crc and timestamp for the uncompressed file shall also be displayed.

For decompression, gzip shall support at least the following compression methods:
- deflate (RFC 1951: DEFLATE Compressed Data Format Specification)
- compress (POSIX 1003.1-2008 (ISO/IEC 9945-2009))

The crc shall be given as ffffffff for a file not in gzip format.

If the --name or -N option is also specified, the uncompressed name, date and time are those stored within the compressed file, if present.

If the --quiet or -q option is also specified, the title and totals lines are not displayed.
-L, --license

displays the gzip license and quit.

-n, --no-name

does not save the original file name and time stamp by default when compressing. (The original name is always saved if the name had to be truncated.) When decompressing, do not restore the original file name if present (remove only the gzip suffix from the compressed file name) and do not restore the original time stamp if present (copy it from the compressed file). This option is the default when decompressing.

-N, --name

always saves the original file name and time stamp when compressing; this is the default. When decompressing, restore the original file name and time stamp if present. This option is useful on systems which have a limit on file name length or when the time stamp has been lost after a file transfer.

-q, --quiet

suppresses all warnings.

-r, --recursive

travels the directory structure recursively. If any of the file names specified on the command line are directories, gzip will descend into the directory and compress all the files it finds there (or decompress them in the case of gunzip).

-S .suf, --suffix .suf

uses suffix .suf instead of .gz.

-t, --test

checks the compressed file integrity.

-v, --verbose

displays the name and percentage reduction for each file compressed or decompressed.

-#, --fast, --best

regulates the speed of compression using the specified digit #, where -1 or --fast indicates the fastest compression method (less compression) and -9 or --best indicates the slowest compression method (best compression). The default compression level is -6 (that is, biased towards high compression at expense of speed).

**LSB Deprecated Options**

The behaviors specified in this section are expected to disappear from a future version of the LSB; applications should only use the non-LSB-deprecated behaviors.

-V, --version

displays the version number and compilation options, then quits.
hostname

Name
hostname — show or set the system’s host name

Synopsis
hostname [name]

Description
hostname is used to either display or, with appropriate privileges, set the current host name of the system. The host name is used by many applications to identify the machine.

When called without any arguments, the program displays the name of the system as returned by the gethostname() function.

When called with a name argument, and the user has appropriate privilege, the command sets the host name.

Note: It is not specified if the hostname displayed will be a fully qualified domain name. Applications requiring a particular format of hostname should check the output and take appropriate action.
install

Name
install — copy files and set attributes

Synopsis
install [option...] SOURCE DEST install [option...] SOURCE... DEST install [-d | --directory] [option...] DIRECTORY...

Description
In the first two formats, copy SOURCE to DEST or multiple SOURCE(s) to the existing DEST directory, optionally setting permission modes and file ownership. In the third format, each DIRECTORY and any missing parent directories shall be created.

Standard Options
--backup[=METHOD]
makes a backup of each existing destination file. METHOD may be one of the following:

none or off
never make backups.

numbered or t
make numbered backups. A numbered backup has the form "%s.~%d~", target_name, version_number. Each backup shall increment the version number by 1.

existing or nil
behave as numbered if numbered backups exist, or simple otherwise.

simple or never
append a suffix to the name. The default suffix is '~', but can be overridden by setting SIMPLE_BACKUP SUFFIX in the environment, or via the -S or --suffix option.

If no METHOD is specified, the environment variable VERSION_CONTROL shall be examined for one of the above. Unambiguous abbreviations of METHOD shall be accepted. If no METHOD is specified, or if METHOD is empty, the backup method shall default to existing.

If METHOD is invalid or ambiguous, install shall fail and issue a diagnostic message.

-b
is equivalent to --backup=existing.

-d, --directory
treats all arguments as directory names; creates all components of the specified directories.

-D
creates all leading components of DEST except the last, then copies SOURCE to
DEST; useful in the 1st format.

-g GROUP, --group=GROUP
  if the user has appropriate privilege, sets group ownership, instead of process' current group. GROUP is either a name in the user group database, or a positive integer, which shall be used as a group-id.

-m MODE, --mode=MODE
  sets permission mode (specified as in chmod), instead of the default rwxr-xr-x.

-o OWNER, --owner=OWNER
  if the user has appropriate privilege, sets ownership. OWNER is either a name in the user login database, or a positive integer, which shall be used as a user-id.

-p, --preserve-timestamps
  copies the access and modification times of SOURCE files to corresponding destination files.

-s, --strip
  strips symbol tables, only for 1st and 2nd formats.

-S SUFFIX, --suffix=SUFFIX
  equivalent to --backup=existing, except if a simple suffix is required, use SUFFIX.

--verbose
  prints the name of each directory as it is created.

-v, --verbose
  print the name of each file before copying it to stdout.

install_initd

Name
install_initd — activate an init script

Synopsis
/usr/lib/lsb/install_initd initd_file

Description
install_initd shall activate a system initialization file that has been copied to an implementation defined location such that this file shall be run at the appropriate point during system initialization. The install_initd command is typically called in the postinstall script of a package, after the script has been copied to /etc/init.d. See also Installation and Removal of Init Scripts.
ipcrm

Name
ipcrm — remove IPC Resources

Synopsis
ipcrm [-q msgid | -Q msgkey | -s semid | -S semkey | -m shmid | -M shmkey]...ipcrm [shm | msg | msg] id...

Description
If any of the -q, -Q, -s, -S, -m, or -M arguments are given, the ipcrm shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009).

Otherwise, ipcrm shall remove the resource of the specified type identified by id.

Future Directions
A future revision of this specification may deprecate the second synopsis form.

Rationale: In its first Linux implementation, ipcrm used the second syntax shown in the SYNOPSIS. Functionality present in other implementations of ipcrm has since been added, namely the ability to delete resources by key (not just identifier), and to respect the same command line syntax. The previous syntax is still supported for backwards compatibility only.
ipcs

Name

ipcs — provide information on ipc facilities

Synopsis

ipcs [-smq] [-tcp]

Description

ipcs provides information on the ipc facilities for which the calling process has read ac-

Note: Although this command has many similarities with the optional ipcs utility described
in POSIX 1003.1-2008 (ISO/IEC 9945-2009), it has substantial differences and is therefore
described separately. The options specified here have similar meaning to those in POSIX
1003.1-2008 (ISO/IEC 9945-2009); other options specified there have unspecified behavior
on an LSB conforming implementation. See Application Usage below. The output format is
not specified.

Resource display options

-m

shared memory segments.

-q

tmessage queues.

-s

semaphore arrays.

Output format options

-t

time.

-p

pid.

-c

creator.

Application Usage

In some implementations of ipcs the -a option will print all information available. In
other implementations the -a option will print all resource types. Therefore, applica-
tions shall not use the -a option.

Some implementations of ipcs provide more output formats than are specified here.
These options are not consistent between differing implementations of ipcs. Therefore,
only the -t, -c and -p option formatting flags may be used. At least one of the -t, -c
and -p options and at least one of -m, -q and -s options shall be specified. If no options
are specified, the output is unspecified.
killall

Name

killall — kill processes by name

Synopsis

killall [-egiqvw] [-signal] name... killall -l killall -V

Description

killall sends a signal to all processes running any of the specified commands. If no signal name is specified, SIGTERM is sent.

Signals can be specified either by name (e.g. -HUP) or by number (e.g. -1). Signal 0 (check if a process exists) can only be specified by number.

If the command name contains a slash (/), processes executing that particular file will be selected for killing, independent of their name.

killall returns a non-zero return code if no process has been killed for any of the listed commands. If at least one process has been killed for each command, killall returns zero.

A killall process never kills itself (but may kill other killall processes).

Standard Options

-e

requires an exact match for very long names. If a command name is longer than 15 characters, the full name may be unavailable (i.e. it is swapped out). In this case, killall will kill everything that matches within the first 15 characters. With -e, such entries are skipped. killall prints a message for each skipped entry if -v is specified in addition to -e.

-g

kills the process group to which the process belongs. The kill signal is only sent once per group, even if multiple processes belonging to the same process group were found.

-i

asks interactively for confirmation before killing.

-l

lists all known signal names.

-q

does not complain if no processes were killed.

-v

reports if the signal was successfully sent.

LSB Deprecated Options

The behaviors specified in this section are expected to disappear from a future version of the LSB; applications should only use the non-LSB-deprecated behaviors.

-V
displays version information.

**lpr**

**Name**

lpr — off line print

**Synopsis**


**Description**

lpr uses a spooling daemon to print the named files when facilities become available. If no names appear, the standard input is assumed.

**Standard Options**

- **-l**
  identifies binary data that is not to be filtered but sent as raw input to printer.

- **-p**
  formats with "pr" before sending to printer.

- **-Pprinter**
  sends output to the printer named printer instead of the default printer.

- **-h**
  suppresses header page.

- **-s**
  uses symbolic links.

- **-#copies**
  specifies copies as the number of copies to print.

- **-J name**
  specifies name as the job name for the header page.

- **-T title**
  specifies title as the title used for "pr".
ls

Name
ls — list directory contents

Description
ls shall behave as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with extensions listed below.

Extensions

-l

If the file is a character special or block special file, the size of the file shall be replaced with two unsigned numbers in the format "%u, %u", representing the major and minor device numbers associated with the special file.

Note: The LSB does not specify the meaning of the major and minor devices numbers.

-p

in addition to POSIX 1003.1-2008 (ISO/IEC 9945-2009) XSI optional behavior of printing a slash for a directory, ls -p may display other characters for other file types.
lsb_release

Name

lsb_release — print distribution specific information

Synopsis

lsb_release [option...]

Description

The lsb_release command prints certain LSB (Linux Standard Base) and Distribution information.

If no options are given, the -v option is the default.

Options

-v, --version

displays the LSB version the distribution is compliant with. The version is expressed as a colon separated list of versioned LSB module identifiers. An LSB module identifier is a dash-separated tuple consisting of module name and version or module name, version and architecture name, in that order. The version output is presented as a single line of text beginning with LSB Version: followed by a tab character, then the list of LSB module identifiers. This format is intended to be easily parsable by programs which need to consume the data. Excepting core, which must always be present, the list of module identifiers may change over time depending on installations and removals of system software.

Example: LSB Version: core-5.0-amd64:core-5.0-noarch:desktop-5.0-amd64:desktop-5.0-noarch:languages-5.0:imaging-5.0

Note: An implementation may support multiple releases of the same module. To claim compliance, the implementation shall list all relevant module identifiers. Version specific library interfaces, if any, will be selected by the program interpreter, which may change from release to release. Version specific commands and utilities, if any, will be described in the relevant specification.

For reporting or querying compliance with this release of the specification, use the module names found in the Package Dependencies section of the Package Format and Installation chapter for that module specification.

-i, --id

displays a string identifying the distribution provider. The id output is a single line of text beginning with Distributor ID: followed by a tab character, then the id string. This specification assigns no meaning to the value of the string, the contents are at the discretion of the distribution provider.

Example: Distributor ID: Frobnozz

-d, --description

displays text describing the distribution. The description output is a single line of text beginning with Description: followed by a tab character, then the description string. This specification assigns no meaning to the value of the string, the contents are at the discretion of the distribution provider.

Example: Description: Frobnozz release 9 (Gilpher)
-r, --release
displays the release number of distribution. The release output is a single line of
text beginning with Release: followed by a tab character, then the release string.
This specification assigns no meaning to the value of the string, the contents are at
the discretion of the distribution provider.

Example: Release: 9

c, --codename
displays a codename which corresponds to the distribution release. The codename
output is a single line of text beginning with Codename: followed by a tab
character, then the codename string. This specification assigns no meaning to the
value of the string, the contents are at the discretion of the distribution provider.

Example: Codename: Gilpher

-a, --all
displays all of the above information.

-s, --short
displays all of the above information in a short output format.

-h, --help
displays a human-readable help message.

m4

Name
m4 — macro processor

Description
m4 is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with extensions as
listed below.

Extensions

-P
forces all builtins to be prefixed with m4_. For example, define becomes
m4_define.

-I directory
Add directory to the end of the search path for includes.
md5sum

Name

md5sum — generate or check MD5 message digests

Synopsis

md5sum [-c [file] | file]

Description

For each file, write to standard output a line containing the MD5 message digest of that file, followed by one or more blank characters, followed by the name of the file. The MD5 message digest shall be calculated according to RFC 1321: The MD5 Message-Digest Algorithm and output as 32 hexadecimal digits.

If no file names are specified as operands, read from standard input and use "-" as the file name in the output.

Options

-c [file]

checks the MD5 message digest of all files named in file against the message digest listed in the same file. The actual format of file is the same as the output of md5sum. That is, each line in the file describes a file. If file is not specified, read message digests from stdin.

Exit Status

md5sum shall exit with status 0 if the sum was generated successfully, or, in check mode, if the check matched. Otherwise, md5sum shall exit with a non-zero status.
mknod

Name
mknod — make special files

Synopsis
mknod [-m mode | --mode=mode] name type [major minor] mknod [--version]

Description
The mknod command shall create a special file named name of the given type.
The type shall be one of the following:

b
creates a block (buffered) special file with the specified major and minor device numbers.

c, u
creates a character (unbuffered) special file with the specified major and minor device numbers.

p
creates a FIFO.

Options

-m mode, --mode=mode
create the special file with file access permissions set as described in mode. The permissions may be any absolute value (i.e. one not containing '+' or '-') acceptable to the chmod command.

--version
output version information and exit.

Note: This option may be deprecated in a future release of this specification.

If type is p, major and minor shall not be specified. Otherwise, these parameters are mandatory.

Future Directions
This command may be deprecated in a future version of this specification. The major and minor operands are insufficiently portable to be specified usefully here. Only a FIFO can be portably created by this command, and the mkfifo command is a simpler interface for that purpose.
mktemp

Name
mktemp — make temporary file name (unique)

Synopsis
mktemp [-q] [-u] template

Description
The mktemp command takes the given file name template and overwrites a portion of it to create a file name. This file name shall be unique and suitable for use by the application.

The template should have at least six trailing 'X' characters. These characters are replaced with characters from the portable filename character set in order to generate a unique name.

If mktemp can successfully generate a unique file name, and the -u option is not present, the file shall be created with read and write permission only for the current user. The mktemp command shall write the filename generated to the standard output.

Options

-q
fail silently if an error occurs. Diagnostic messages to stderr are suppressed, but the command shall still exit with a non-zero exit status if an error occurs.

-u
operates in `unsafe' mode. A unique name is generated, but the temporary file shall be unlinked before mktemp exits. Use of this option is not encouraged.
**more**

**Name**

more — display files on a page-by-page basis

**Description**

more is as specified in [POSIX 1003.1-2008 (ISO/IEC 9945-2009)](http://pubs.opengroup.org/onlinepubs/9699919799/utilities/more.html), but with differences as listed below.

**Differences**

The more command need not respect the LINES and COLUMNS environment variables.

The following additional options may be supported:

- **-num**
  
  specifies an integer which is the screen size (in lines).

- **+num**

  starts at line number num.

- **+/pattern**

  Start at the first line matching the pattern, equivalent to executing the search forward (/) command with the given pattern immediately after opening each file.

The following options from [POSIX 1003.1-2008 (ISO/IEC 9945-2009)](http://pubs.opengroup.org/onlinepubs/9699919799/utilities/more.html) may behave differently:

- **-e**

  has unspecified behavior.

- **-i**

  has unspecified behavior.

- **-n**

  has unspecified behavior.

- **-p**

  Either clear the whole screen before displaying any text (instead of the usual scrolling behavior), or provide the behavior specified by [POSIX 1003.1-2008 (ISO/IEC 9945-2009)](http://pubs.opengroup.org/onlinepubs/9699919799/utilities/more.html). In the latter case, the syntax is "-p command".

- **-t**

  has unspecified behavior.

The more command need not support the following interactive commands:
Rationale

The `+num` and `+/string` options are deprecated in SUSy2, and have been removed in POSIX 1003.1-2008 (ISO/IEC 9945-2009); however this specification continues to specify them because the publicly available util-linux package does not support the replacement (`-p command`). The `+command` option as found in SUSy2 is more general than is specified here, but the util-linux package appears to only support the more specific `+num` and `+/string` forms.
mount

Name

mount — mount a file system

Synopsis


Description

As described in POSIX 1003.1-2008 (ISO/IEC 9945-2009), all files in the system are organized in a directed graph, known as the file hierarchy, rooted at / . These files can be spread out over several underlying devices. The mount command shall attach the file system found on some underlying device to the file hierarchy.

Options

- v

invoke verbose mode. The mount command shall provide diagnostic messages on stdout.

- a

mount all file systems (of the given types) mentioned in /etc/fstab.

- F

If the - a option is also present, fork a new incarnation of mount for each device to be mounted. This will do the mounts on different devices or different NFS servers in parallel.

- f

cause everything to be done except for the actual system call; if it's not obvious, this `fakes' mounting the file system.

- n

mount without writing in /etc/mtab. This is necessary for example when /etc is on a read-only file system.

- s

ignore mount options not supported by a file system type. Not all file systems support this option.

- r

mount the file system read-only. A synonym is -o ro.

- w

mount the file system read/write. (default) A synonym is -o rw.

- L label

If the file /proc/partitions is supported, mount the partition that has the specified label.

- U uuid

© 2015 Linux Foundation
If the file `/proc/partitions` is supported, mount the partition that has the specified uuid.

```
-t vfstype
```

indicating a file system type of `vfstype`.

More than one type may be specified in a comma separated list. The list of file system types can be prefixed with `no` to specify the file system types on which no action should be taken.

```
-o
```

options are specified with a `-o` flag followed by a comma-separated string of options. Some of these options are only useful when they appear in the `/etc/fstab` file. The following options apply to any file system that is being mounted:

- `async`
  - perform all I/O to the file system asynchronously.
- `atime`
  - update inode access time for each access. (default)
- `auto`
  - in `/etc/fstab`, indicate the device is mountable with `-a`.
- `defaults`
  - use default options: `rw, suid, dev, exec, auto, nouser, async`.
- `dev`
  - interpret character or block special devices on the file system.
- `exec`
  - permit execution of binaries.
- `noatime`
  - do not update file access times on this file system.
- `noauto`
  - in `/etc/fstab`, indicates the device is only explicitly mountable.
- `nodev`
  - do not interpret character or block special devices on the file system.
- `noexec`
  - do not allow execution of any binaries on the mounted file system.
- `nosuid`
  - do not allow set-user-identifier or set-group-identifier bits to take effect.
- `nouser`
  - forbid an unprivileged user to mount the file system. (default)
- `remount`
LSB Core - Generic 5.0

remount an already-mounted file system. This is commonly used to change
the mount options for a file system, especially to make a read-only file system
writable.

ro

mount the file system read-only.

rw

mount the file system read-write.

suid

allow set-user-identifier or set-group-identifier bits to take effect.

sync

do all I/O to the file system synchronously.

user

allow an unprivileged user to mount the file system. This option implies the
options noexec, nosuid, nodev unless overridden by subsequent options.

LSB Deprecated Options

The behaviors specified in this section are expected to disappear from a future version
of the LSB; applications should only use the non-LSB-deprecated behaviors.

-V

output version and exit.
msgfmt

Name

msgfmt — create a message object from a message file

Synopsis

msgfmt [options...] filename...

Description

The **msgfmt** command generates a binary message catalog from a textual translation description. Message catalogs, or message object files, are stored in files with a .mo extension.

**Note:** The format of message object files is not guaranteed to be portable. Message catalogs should always be generated on the target architecture using the `msgfmt` command.

The source message files, otherwise known as portable object files, have a .po extension.

The **filename** operands shall be portable object files. The .po file contains messages to be displayed to users by system utilities or by application programs. The portable object files are text files, and the messages in them can be rewritten in any language supported by the system.

If any **filename** is -, a portable object file shall be read from the standard input.

The **msgfmt** command interprets data as characters according to the current setting of the LC_CTYPE locale category.

Options

- `-c`  
  --check

  Detect and diagnose input file anomalies which might represent translation errors. The **msgid** and **msgstr** strings are studied and compared. It is considered abnormal that one string starts or ends with a newline while the other does not.

  If the message is flagged as c-format (see **Comment Handling**), check that the **msgid** string and the **msgstr** translation have the same number of % format specifiers, with matching types.

- `-D directory`  
  --directory=directory

  Add directory to list for input files search. If **filename** is not an absolute pathname and **filename** cannot be opened, search for it in **directory**. This option may be repeated. Directories shall be searched in order, with the leftmost **directory** searched first.

- `-f`  
  --use-fuzzy

  Use entries marked as fuzzy in output. If this option is not specified, such entries are not included into the output. See **Comment Handling** below.

- `-o output-file`  
  --output-file=output-file
LSB Core - Generic 5.0

Specify the output file name as output-file. If multiple domains or duplicate msgids in the .po file are present, the behavior is unspecified. If output-file is -, output is written to standard output.

--strict

Ensure that all output files have a .mo extension. Output files are named either by the -o (or --output-file) option, or by domains found in the input files.

-v
--verbose

Print additional information to the standard error, including the number of translated strings processed.

Operands

The filename operands are treated as portable object files. The format of portable object files is defined in EXTENDED DESCRIPTION.

Standard Input

The standard input is not used unless a filename operand is specified as "-".

Environment Variables

LANGUAGE

Specifies one or more locale names.

LANG

Specifies locale name.

LC_ALL

Specifies locale name for all categories. If defined, overrides LANG, LC_CTYPE and LC_MESSAGES.

LC_CTYPE

Determine the locale for the interpretation of sequences of bytes of text data as characters (for example, single-byte as opposed to multi-byte characters in arguments and input files).

LC_MESSAGES

Specifies messaging locale, and if present overrides LANG for messages.

Standard Output

The standard output is not used unless the option-argument of the -o option is specified as "-".

Extended Description

The format of portable object files (.po files) is defined as follows. Each .po file contains one or more lines, with each line containing either a comment or a statement. Comments start the line with a hash mark (#) and end with the newline character. Empty lines, or lines containing only white-space, shall be ignored. Comments can in certain circumstances alter the behavior of msgfmt. See Comment Handling below for details on comment processing. The format of a statement is:

directive value
Each directive starts at the beginning of the line and is separated from value by white space (such as one or more space or tab characters). The value consists of one or more quoted strings separated by white space. If two or more strings are specified as value, they are normalized into single string using the string normalization syntax specified in ISO C (1999). The following directives are supported:

```
domain domainname
msgid message_identifier
msgid_plural untranslated_string_plural
msgstr message_string
msgstr[n] message_string
```

The behavior of the domain directive is affected by the options used. See OPTIONS for the behavior when the -o option is specified. If the -o option is not specified, the behavior of the domain directive is as follows:

1. All msgids from the beginning of each .po file to the first domain directive are put into a default message object file, messages (or messages.mo if the --strict option is specified).
2. When msgfmt encounters a domain domainname directive in the .po file, all following msgids until the next domain directive are put into the message object file domainname (or domainname.mo if --strict option is specified).
3. Duplicate msgids are defined in the scope of each domain. That is, a msgid is considered a duplicate only if the identical msgid exists in the same domain.
4. All duplicate msgids are ignored.

The msgid directive specifies the value of a message identifier associated with the directive that follows it. The msgid_plural directive specifies the plural form message specified to the plural message handling functions ngettext(), dngettext() or dcngettext(). The message_identifier string identifies a target string to be used at retrieval time. Each statement containing a msgid directive shall be followed by a statement containing a msgstr directive or msgstr[n] directives.

The msgstr directive specifies the target string associated with the message_identifier string declared in the immediately preceding msgid directive.

The msgstr[n] (where n = 0, 1, 2, ...) directive specifies the target string to be used with plural form handling functions ngettext(), dngettext() and dcngettext().

Message strings can contain the following escape sequences:

**Table 17-1 Escape Sequences**

© 2015 Linux Foundation
| \n     | newline          |
| \t    | tab              |
| \v    | vertical tab     |
| \b    | backspace        |
| \r    | carriage return  |
| \f    | formfeed         |
| \\   | backslash        |
| \"   | double quote     |
| \ddd  | octal bit pattern|
Comment Handling

Comments are introduced by a #, and continue to the end of the line. The second character (i.e. the character following the #) has special meaning. Regular comments should follow a space character. Other comment types include:

- # normal-comments
- #. automatic-comments
- #: reference...
- #, flag

Automatic and reference comments are typically generated by external utilities, and are not specified by the LSB. The `msgfmt` command shall ignore such comments.

**Note:** Portable object files may be produced by unspecified tools. Some of the comment types described here may arise from the use of such tools. It is beyond the scope of this specification to describe these tools.

The #, comments require one or more flags separated by the comma (,) character. The following flags can be specified:

- **fuzzy**
  
  This flag shows that the following msgstr string might not be a correct translation. Only the translator (i.e. the individual undertaking the translation) can judge if the translation requires further modification, or is acceptable as is. Once satisfied with the translation, the translator then removes this fuzzy flag.

  If this flag is specified, the `msgfmt` utility will not generate the entry for the immediately following `msgid` in the output message catalog, unless the `-u use-fuzzy` is specified.

- **c-format**
- **no-c-format**

  The `c-format` flag indicates that the `msgid` string is used as format string by `printf()`-like functions. If the `c-format` flag is given for a string the `msgfmt` utility may perform additional tests to check the validity of the translation.

Plurals

The `msgid` entry with empty string ("") is called the header entry and is treated specially. If the message string for the header entry contains `nplurals=value`, the value indicates the number of plural forms. For example, if `nplurals=4`, there are 4 plural forms. If `nplurals` is defined, there should be a `plural=expression` on the same line, separated by a semicolon (;) character. The expression is a C language expression to determine which version of `msgstr[n]` to be used based on the value of `n`, the last argument of `ngettext()`, `dngettext()` or `dcngettext()`. For example:

```
nplurals=2; plural=n == 1 ? 0 : 1
```

indicates that there are 2 plural forms in the language; `msgstr[0]` is used if `n == 1`, otherwise `msgstr[1]` is used. Another example:

```
nplurals=3; plural=n==1 ? 0 : n==2 ? 1 : 2
```

indicates that there are 3 plural forms in the language; `msgstr[0]` is used if `n == 1`, `msgstr[1]` is used if `n == 2`, otherwise `msgstr[2]` is used.
If the header entry contains charset=\texttt{codeset} string, the \texttt{codeset} is used to indicate the codeset to be used to encode the message strings. If the output string's codeset is different from the message string's codeset, codeset conversion from the message string's codeset to the output string's codeset will be performed upon the call of \texttt{gettext()}, \texttt{dgettext()}, \texttt{dcgettext()}, \texttt{ngettext()}, \texttt{dngettext()}, and \texttt{dcngettext}(). The output string's codeset is determined by the current locale's codeset (the return value of \texttt{nl_langinfo(CODESET)}) by default, and can be changed by the call of \texttt{bind_textdomain_codeset()}. 

**Exit Status**

The following exit values are returned:

0

  Successful completion.

>0

  An error occurred.

**Application Usage**

Neither \texttt{msgfmt} nor any \texttt{gettext()} function imposes a limit on the total length of a message. Installing message catalogs under the C locale is pointless, since they are ignored for the sake of efficiency.

**Examples**

Example 1: Examples of creating message objects from message files.
In this example \texttt{module1.po}, \texttt{module2.po} and \texttt{module3.po} are portable message object files.

```
example% cat module1.po
# default domain "messages"
msgid "message one"
msgstr "mensaje número uno"
#
domain "help_domain"
msgid "help two"
msgstr "ayuda número dos"
#
domain "error_domain"
msgid "error three"
msgstr "error número tres"
```

```
example% cat module2.po
# default domain "messages"
msgid "message four"
```
The following command will produce the output files messages, help_domain, and error_domain.

eexample% msgfmt module1.po

The following command will produce the output files messages.mo, help_domain.mo, error_domain.mo, and window_domain.mo.

eexample% msgfmt module1.po module2.po

The following example will produce the output file hello.mo.

eexample% msgfmt -o hello.mo module3.po

**newgrp**

**Name**

newgrp — change group ID

**Synopsis**

e**newgrp** [group]

**Description**


**Differences**

od

Name
od — dump files in octal and other formats

Synopsis
od [-abcdfilox] [-w width | --width=width] [-v] [-A address_base] [-j skip]
[-n count] [-t type_string] [file...] od --traditional [options] [file]
[[+]offset [.b]] [[+]label [.b]]

Description
The od command shall provide all of the mandatory functionality specified in POSIX
1003.1-2008 (ISO/IEC 9945-2009), but with extensions and differences to the XSI op-
tional behavior as listed below.

Extensions and Differences
-s
unspecified behavior.

Note: Applications wishing to achieve the POSIX 1003.1-2008 (ISO/IEC 9945-2009)
behavior for -s should instead use -t d2.

-w width, --width=width
each output line is limited to width bytes from the input.

--traditional
accepts arguments in traditional form, see Traditional Usage below.

Note: The XSI optional behavior for offset handling described in POSIX 1003.1-2008
(ISO/IEC 9945-2009) is not supported unless the --traditional option is also
specified.

Pre-POSIX and XSI Specifications
The LSB supports mixing options between the mandatory and XSI optional synopsis
forms in POSIX 1003.1-2008 (ISO/IEC 9945-2009). The LSB shall support the follow-
ing options:

-a
is equivalent to -t a, selects named characters.

-b
is equivalent to -t o1, selects octal bytes.

-c
is equivalent to -t c, selects characters.

-d
is equivalent to -t u2, selects unsigned decimal two byte units.

-f
is equivalent to -t fF, selects floats.
-i

is equivalent to -t d2, selects decimal two byte units.

Note: This usage may change in future releases; portable applications should use -t d2.

-l

is equivalent to -t d4, selects decimal longs.

-o

is equivalent to -t o2, selects octal two byte units.

-x

is equivalent to -t x2, selects hexadecimal two byte units.

Note that the XSI option -s need not be supported.

Traditional Usage

If the --traditional option is specified, there may be between zero and three operands specified.

If no operands are specified, then od shall read the standard input.

If there is exactly one operand, and it is an offset of the form [+]+offset[.]b, then it shall be interpreted as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009). The file to be dumped shall be the standard input.

If there are exactly two operands, and they are both of the form [+]+offset[.]b, then the first shall be treated as an offset (as above), and the second shall be a label, in the same format as the offset. If a label is specified, then the first output line produced for each input block shall be preceded by the input offset, cumulative across input files, of the next byte to be written, followed by the label, in parentheses. The label shall increment in the same manner as the offset.

If there are three operands, then the first shall be the file to dump, the second the offset, and the third the label.

Note: Recent versions of coreutils contain an od utility that conforms to POSIX 1003.1-2008 (ISO/IEC 9945-2009). However, in April 2005, this version was not in widespread use. A future version of this specification may remove the differences.
passwd

Name
passwd — change user password

Synopsis
passwd [-x max] [-n min] [-w warn] [-i inactive] name passwd {-l | -u} name

Description
passwd changes authentication information for user and group accounts, including passwords and password expiry details, and may be used to enable and disable accounts. Only a user with appropriate privilege may change the password for other users or modify the expiry information.

Options
-x max
sets the maximum number of days a password remains valid.

-n min
sets the minimum number of days before a password may be changed.

-w warn
sets the number of days warning the user will receive before their password will expire.

-i inactive
disables an account after the password has been expired for the given number of days.

-l
disables an account by changing the password to a value which matches no possible encrypted value.

-u
re-enables an account by changing the password back to its previous value.
patch

Name
patch — apply a diff file to an original

Description
patch is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with extensions as listed below.

Extensions
--binary
reads and write all files in binary mode, except for standard output and /dev/tty. This option has no effect on POSIX-compliant systems.

-u, --unified
interprets the patch file as a unified context diff.

pidof

Name
pidof — find the process ID of a running program

Synopsis
pidof [-s] [-x] [-o omitpid...] program...

Description
Return the process ID of a process which is running the program named on the command line.

The pidof command is a system administration utility, see Path For System Administration Utilities.

Options
-s
instructs the program to only return one pid.

-x
causes the program to also return process id's of shells running the named scripts.

-o
omits processes with specified process id.
remove_initd

**Name**
remove_initd — clean up init script system modifications introduced by install_initd

**Synopsis**
/usr/lib/lsb/remove_initd initd_file

**Description**
remove_initd processes the removal of the modifications made to a distribution's init script system by the install_initd program. This cleanup is performed in the preuninstall script of a package; however, the package manager is still responsible for removing the script from the repository. See also Installation and Removal of Init Scripts.

renice

**Name**
renice — alter priority of running processes

**Description**
renice is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

**Differences**
- `n increment` has unspecified behavior.

sed

**Name**
sed — stream editor

**Description**
sed is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences as listed below.

**LSB Differences**
Certain aspects of internationalized regular expressions are optional; see Regular Expressions.
sendmail

Name
sendmail — an electronic mail transport agent

Synopsis
/usr/sbin/sendmail [options] [address...]

Description
To deliver electronic mail (email), implementations shall provide the interface described here as a command named sendmail. This interface shall be the default delivery method for applications needing to transmit an email message.

This program sends an email message to one or more recipients, routing the message as necessary. This program is not intended as a user interface routine.

With no options, sendmail reads its standard input up to an end-of-file or a line consisting only of a single dot and sends a copy of the message found there to all of the addresses listed. It determines the network(s) to use based on the syntax and contents of the addresses.

If an address is preceded by a backslash, '\', it is unspecified whether the address is subject to local alias expansion.

The format of messages shall be as defined in RFC 2822:Internet Message Format.

Note: The required name sendmail was chosen for historical reasons, but the behaviors specified here are intended to reflect functionality which can be provided by a number of other implementations such as smail, exim and others. This specification does not require that the specific sendmail program be the implementation chosen to provide this functionality.

Options
-bm
read mail from standard input and deliver it to the recipient addresses. This is the default mode of operation.

-bp
If the user has sufficient privilege, list information about messages currently in the mail queue.

-bs
use the SMTP protocol as described in RFC 2821:Simple Mail Transfer Protocol; read SMTP commands on standard input and write SMTP responses on standard output.

In this mode, sendmail shall accept \r\n (CR-LF), as required by RFC 2821:Simple Mail Transfer Protocol, and \n (LF) line terminators.

-F fullname
explicitly set the full name of the sender for incoming mail unless the message already contains a From: message header.

If the user running sendmail is not sufficiently trusted, then the actual sender may be indicated in the message, depending on the configuration of the agent.

-f name
explicitly set the envelope sender address for incoming mail. If there is no From: header, the address specified in the From: header will also be set.

If the user running sendmail is not sufficiently trusted, then the actual sender shall be indicated in the message.

-i
ignore dots alone on lines by themselves in incoming messages. If this options is not specified, a line consisting of a single dot shall terminate the input. If -bs is also used, the behavior is unspecified.

-odb
deliver any mail in background, if supported; otherwise ignored.

-odf
deliver any mail in foreground, if supported; otherwise ignored.

-oem or -em
mail errors back to the sender. (default)

-oep or -ep
write errors to the standard error output.

-oeq or -eq
do not send notification of errors to the sender. This only works for mail delivered locally.

-oi
is equivalent to -i.

-om
indicate that the sender of a message should receive a copy of the message if the sender appears in an alias expansion. Ignored if aliases are not supported.

-t
read the message to obtain recipients from the To:, Cc:, and Bcc: headers in the message instead of from the command arguments. If a Bcc: header is present, it is removed from the message unless there is no To: or Cc: header, in which case a Bcc: header with no data is created, in accordance with RFC 2822:Internet Message Format.

If there are any operands, the recipients list is unspecified.
This option may be ignored when not in -bm mode (the default).

Note: It is recommended that applications use as few options as necessary, none if possible.

Exit status
The sendmail command returns an exit status indicating the results of the operation. The exit codes are as defined in <sys/exits.h>.

EX_OK
successful completion on all addresses. Note this does not necessarily indicate successful delivery.

EX_NOUSER
The user specified did not exist.

EX_UNAVAILABLE
A required service or resource was unavailable.

EX_USAGE
Command was invoked incorrectly, such as wrong number of arguments, syntax error in arguments, bad flags.

EX SOFTWARE
An internal software error was detected. Includes bad arguments.

EX OSERROR
An operating system error was detected. Includes failure to execute fork() or pipe().

EX_NOHOST
The host specified did not exist.

EX TEMPFAIL
Temporary failure. Used to indicate the message could not be sent immediately, perhaps because a connection could not be created, but the request can be retried.
**seq**

**Name**

seq — generate a sequence of numbers

**Synopsis**

/usr/bin/seq [-f fmt_str] [-s sep_str] [first_num] [inc_num] last_num

**Description**

The seq command shall output a sequence of numbers from first_num to last_num, stepping by the increment inc_num. The first_num and last_num parameters may be omitted, and default to 1 even when first_num is greater than last_num. Floating-point values may be specified for first_num, inc_num, and last_num.

The fmt_str parameter is a floating point format string like the one used for the printf() function in C.

The sep_str parameter string separates the values that are output. The default is a newline character (\n).

**Note:** If first_num is less than last_num and inc_num is negative, or first_num is greater than last_num and inc_num is positive, seq shall not generate any output.

**Standard Options**

- **-f fmt_str**
  Format the numbers in the output sequence according to fmt_str, a floating point format string like the one used for the printf() function in C.

- **-s sep_str**
  Separate the numbers in the output sequence with sep_str. The default separator string is a newline character (\n).

**first_num**

The first number in the output sequence. Defaults to 1. May be a floating point value.

**inc_num**

The increment for the output sequence. Defaults to 1. May be a floating point value.

**last_num**

The last number in the output sequence. May be a floating point value.
**sh**

**Name**

sh — shell, the standard command language interpreter

**Description**


**Shell Invocation**

The shell shall support an additional option, `-l` (the letter *ell*). If the `-l` option is specified, or if the first character of argument zero (the command name) is a `' '-'`, this invocation of the shell is a *login shell*.

An interactive shell, as specified in [POSIX 1003.1-2008 (ISO/IEC 9945-2009)](https://www.opengroup.org/onlinepubs/007908487/xsh/sh.html), that is also a login shell, or any shell if invoked with the `-l` option, shall, prior to reading from the input file, first read and execute commands from the file `/etc/profile`, if that file exists, and then from a file called `~/.profile`, if such a file exists.

**Note:** This specification requires that the sh utility shall also read and execute commands in its current execution environment from all the shell scripts in the directory `/etc/profile.d`. Such scripts are read and executed as a part of reading and executing `/etc/profile`. 
shutdown

Name

shutdown — shut the system down

Synopsis

/sbin/shutdown [-t sec] [-h | -r] [-akfF] time [warning-message]
/sbin/shutdown -c [warning-message]

Description

The shutdown command shall shut the system down in a secure way (first synopsis), or cancel a pending shutdown (second synopsis). When the shutdown is initiated, all logged-in users shall be notified immediately that the system is going down, and users shall be prevented from logging in to the system. The time specifies when the actual shutdown shall commence. See below for details. At the specified time all processes are first notified that the system is going down by the signal SIGTERM. After an interval (see -t) all processes shall be sent the signal SIGKILL. If neither the -h or the -r argument is specified, then the default behavior shall be to take the system to a runlevel where administrative tasks can be run. See also Run Levels.

Note: This is sometimes referred to as “single user mode”.

The -h and -r options are mutually exclusive. If either the -h or -r options are specified, the system shall be halted or rebooted respectively.

Standard Options

-a

use access control. See below.

-t sec

tell the system to wait sec seconds between sending processes the warning and the kill signal, before changing to another runlevel. The default period is unspecified.

-k

do not really shutdown; only send the warning messages to everybody.

-r

reboot after shutdown.

-h

halt after shutdown. Actions after halting are unspecified (e.g. power off).

-f

advise the system to skip file system consistency checks on reboot.

-F

advise the system to force file system consistency checks on reboot.

-c

cancel an already running shutdown.

time
specify when to shut down.

The time argument shall have the following format: [\texttt{now} | [+\texttt{mins} | \texttt{hh:mm}]
If the format is \texttt{hh:mm}, \texttt{hh} shall specify the hour (1 or 2 digits) and \texttt{mm} is the minute of the hour (exactly two digits), and the shutdown shall commence at the next occurrence of the specified time. If the format is \texttt{mins} (or +\texttt{mins}), where \texttt{mins} is a decimal number, shutdown shall commence in the specified number of minutes.

The word \texttt{now} is an alias for +0.

\textbf{warning-message}

specify a message to send to all users.

\textbf{Access Control}

If the \texttt{shutdown} utility is invoked with the -a option, it shall check that an authorized user is currently logged in on the system console. Authorized users are listed, one per line, in the file \texttt{/etc/shutdown.allow}. Lines in this file that begin with a '#' or are blank shall be ignored.

\textbf{Note}: The intent of this scheme is to allow a keyboard sequence entered on the system console (e.g. CTRL-ALT-DEL, or STOP-A) to automatically invoke \texttt{shutdown -a}, and can be used to prevent unauthorized users from shutting the system down in this fashion.
su

Name
su — change user ID

Synopsis
su [options] [-] [username [ARGS]]

Description
The su command shall start a shell running with the real and effective user and group IDs of the user username. If username is not specified, su shall default to an unspecified user with all appropriate privileges. If the -s or --shell is not specified, the shell to be invoked shall be that specified for username in the user database (see getpwnam()), or /bin/sh if there is no shell specified in the user database.

If the - option is specified, or if the first operand is -, the environment for the shell shall be initialized as if the new shell was a login shell (see Shell Invocation).

If the invoking user does not have appropriate privileges, the su command shall prompt for a password and validate this before continuing. Invalid passwords shall produce an error message. The su command shall log in an unspecified manner all invokations, whether successful or unsuccessful.

Any operands specified after the username shall be passed to the invoked shell.

If the option - is not specified, and if the first operand is not -, the environment for the new shell shall be initialized from the current environment. If none of the -m, -p, or --preserve-environment options are specified, the environment may be modified in unspecified ways before invoking the shell. If any of the -m, -p, or --preserve-environment options are specified, the environment shall not be altered.

Note: Although the su command shall not alter the environment, the invoked shell may still alter it before it is ready to interpret any commands.

Standard Options

-the invoked shell shall be a login shell.

-c command, --command=command
Invoke the shell with the option -c command.

-m, -p, --preserve-environment
The current environment shall be passed to the invoked shell. If the environment variable SHELL is set, it shall specify the shell to invoke, if it matches an entry in /etc/shells. If there is no matching entry in /etc/shells, this option shall be ignored if the - option is also specified, or if the first operand is -.

-s shell, --shell=shell
Invoke shell as the command interpreter. The shell specified shall be present in /etc/shells.
sync

Name
sync — flush file system buffers

Synopsis
sync

Description
Force changed blocks to disk, update the super block.

tar

Name
tar — file archiver

Description
tar is as specified in SUSv2, but with differences as listed below.

Differences
Some elements of the Pattern Matching Notation are optional; see Pattern Matching Notation.

- h
  doesn't dump symlinks; dumps the files they point to.

- z
  filters the archive through gzip.
umount

Name

umount — unmount file systems

Synopsis


Description

umount detaches the file system(s) mentioned from the file hierarchy. A file system is specified by giving the directory where it has been mounted.

Standard Options

-v

invokes verbose mode.

-n

unmounts without writing in /etc/mtab.

-r

tries to remount read-only if unmounting fails.

-a

unmounts all of the file systems described in /etc/mtab except for the proc file system.

-t vfstype

indicates that the actions should only be taken on file systems of the specified type. More than one type may be specified in a comma separated list. The list of file system types can be prefixed with no to specify the file system types on which no action should be taken.

-f

forces unmount (in case of an unreachable NFS system).

LSB Deprecated Options

The behaviors specified in this section are expected to disappear from a future version of the LSB; applications should only use the non-LSB-deprecated behaviors.

-V

print version and exits.
useradd

Name
useradd — create a new user or update default new user information

Synopsis
useradd [-c comment] [-d home_dir] [-g initial_group] [-G group...] [-m [-k skeleton_dir]] [-p passwd] [-r] [-s shell] [-u uid [ -o]] login useradd
-D [-g default_group] [-b default_home] [-s default_shell]

Description
When invoked without the -D option, and with appropriate privilege, useradd creates a
new user account using the values specified on the command line and the default values
from the system. The new user account will be entered into the system files as needed,
the home directory will be created, and initial files copied, depending on the command
line options. If the group named group does not exist, or the user names login already
exists, or the user ID uid is already in use and no -o option is specified, then useradd
shall issue a diagnostic message and exit with a non-zero exit status.

When invoked with the -D option, useradd will either display the current default val-
ues, or, with appropriate privilege, update the default values from the command line. If
no options are specified, useradd displays the current default values.

The useradd command is a system administration utility, see Path For System
Administration Utilities.

Standard Options
-c comment
    specifies the new user's password file comment field value.

-d home_dir
    creates the new user using home_dir as the value for the user's login directory. The
default is to append the login name to default_home and use that as the login
directory name.

-g initial_group
    specifies the group name or number of the user's initial login group. The group
name shall exist. A group number shall refer to an already existing group. If -g is
not specified, the implementation will follow the normal user default for that
system. This may create a new group or choose a default group that normal users
are placed in. Applications which require control of the groups into which a user is
placed should specify -g.

-G group[,...]
    specifies a list of supplementary groups which the user is also a member of. Each
group is separated from the next by a comma, with no intervening whitespace. The
groups are subject to the same restrictions as the group given with the -g option.
The default is for the user to belong only to the initial group.

-m [-k skeleton_dir]
LSB Core - Generic 5.0

specifies the user's home directory will be created if it does not exist. The files contained in skeleton_dir will be copied to the home directory if the -k option is used, otherwise the files contained in /etc/skel will be used instead. Any directories contained in skeleton_dir or /etc/skel will be created in the user's home directory as well. The -k option is only valid in conjunction with the -m option. The default is to not create the directory and to not copy any files.

-p passwd
is the encrypted password, as returned by crypt(). The default is to disable the account.

-r
creates a system account, that is, a user with a User ID in the range reserved for system account users. If there is not a User ID free in the reserved range the command will fail.

-s shell
specifies the name of the user's login shell. The default is to leave this field blank, which causes the system to select the default login shell.

-u uid [-o]
specifies the numerical value of the user's ID. This value shall be unique, unless the -o option is used. The value shall be non-negative. The default is the smallest ID value greater than 499 which is not yet used.

Change Default Options

-b default_home
specifies the initial path prefix for a new user's home directory. The user's name will be affixed to the end of default_home to create the new directory name if the -d option is not used when creating a new account.

-g default_group
specifies the group name or ID for a new user's initial group. The named group shall exist, and a numerical group ID shall have an existing entry.

-s default_shell
specifies the name of the new user's login shell. The named program will be used for all future new user accounts.

-c comment
specifies the new user's password file comment field value.

Application Usage

The -D option will typically be used by system administration packages. Most applications should not change defaults which will affect other applications and users.
userdel

Name
userdel — delete a user account and related files

Synopsis
userdel [-r] login

Description
Delete the user account named login. If there is also a group named login, this command may delete the group as well, or may leave it alone. If the user named login does not exist or is currently logged in, userdel shall issue a diagnostic message and exit with a non-zero exit status.

The userdel command is a system administration utility, see Path For System Administration Utilities.

Options
-r
removes files in the user's home directory along with the home directory itself. Files located in other file system will have to be searched for and deleted manually.
usermod

Name
usermod — modify a user account

Synopsis
usermod [-c comment] [-d home_dir [ -m]] [-g initial_group] [-G group [,...]] [-l login_name] [-p passwd] [-s shell] [-u uid [ -o]] login

Description
The usermod command shall modify an entry in the user account database.

The usermod command is a system administration utility, see Path For System Administration Utilities.

Options
-c comment
specifies the new value of the user's password file comment field.

-d home_dir
specifies the user's new login directory. If the -m option is given the contents of the current home directory will be moved to the new home directory, which is created if it does not already exist.

-g initial_group
specifies the group name or number of the user's new initial login group. The group name shall exist. A group number shall refer to an already existing group.

-G group,[...]
specifies a list of supplementary groups which the user is also a member of. Each group is separated from the next by a comma, with no intervening whitespace. The groups are subject to the same restrictions as the group given with the -g option. If the user is currently a member of a group which is not listed, the user will be removed from the group.

-l login_name
changes the name of the user from login to login_name. Nothing else is changed. In particular, the user's home directory name should probably be changed to reflect the new login name.

-p passwd
is the encrypted password, as returned by crypt(3).

-s shell
specifies the name of the user's new login shell. Setting this field to blank causes the system to select the default login shell.

-u uid [ -o]
specifies the numerical value of the user's ID. This value shall be unique, unless the
-o option is used. The value shall be non-negative. Any files which the user owns
and which are located in the directory tree rooted at the user's home directory will
have the file user ID changed automatically. Files outside of the user's home
directory shall be altered manually.

**xargs**

**Name**

*xargs* — build and execute command lines from standard input

**Description**

*xargs* is as specified in POSIX 1003.1-2008 (ISO/IEC 9945-2009), but with differences
as listed below.

**Differences**

-E

has unspecified behavior.

-I

has unspecified behavior.

-L

has unspecified behavior.

**Note:** These options have been implemented in findutils-4.2.9, but this version of the utilit-
ies is not in widespread use as of April 2005. However, future versions of this specification
will require support for these arguments.

**zcat**

**Name**

*zcat* — uncompress files to standard output

**Description**

The *zcat* utility shall behave as described in POSIX 1003.1-2008 (ISO/IEC 9945-2009),
with differences listed below.

The Filesystem Hierarchy Standard requires that if *zcat* exists, it must be a symbolic or
hard link to /bin/gzip. This specification additionally allows *zcat* to be a wrapper
script which calls *gzip -c -d*.

**Differences**

The *zcat* utility shall write to standard output the uncompressed form of files that have
been compressed using any of the compression methods supported by the *gzip* utility. It
is the equivalent of *gzip -c -d*. Input files are not affected.
VII Execution Environment
18 File System Hierarchy

An LSB conforming implementation shall provide the mandatory portions of the file system hierarchy specified in the Filesystem Hierarchy Standard (FHS), together with any additional requirements made in this specification.

An LSB conforming application shall conform to the Filesystem Hierarchy Standard.

The FHS allows many components or subsystems to be optional. An application shall check for the existence of an optional component before using it, and should behave in a reasonable manner if the optional component is not present.

The FHS requirement to locate the operating system kernel in either / or /boot does not apply if the operating system kernel does not exist as a file in the file system.

The FHS specifies certain behaviors for a variety of commands if they are present (for example, ping or python). However, conforming applications shall not rely on any commands beyond those required by this specification. The mere existence of a command may not be used as an indication that the command behaves in any particular way.

The following directories or links need not be present: /etc/X11 /usr/bin/X11 /usr/lib/X11 /proc

18.1 /dev: Device Files

The devices described in Chapter 6. "Operating System Specific Annex", Section 6.1. "Linux", subsection 6.1.3. "(dev): Devices and special files" in the Filesystem Hierarchy Standard are required on an LSB conforming system. Other devices may also exist in /dev. Device names may exist as symbolic links to other device nodes located in /dev or subdirectories of /dev. There is no requirement concerning major/minor number values.

18.2 /etc: Host-specific system configuration

In addition to the requirements for /etc in the Filesystem Hierarchy Standard, an LSB conforming system shall also provide the following directories or symbolic links to directories:

/etc/cron.d
A directory containing extended crontab files; see Cron Jobs.

/etc/cron.daily
A directory containing shell scripts to be executed once a day; see Cron Jobs.

/etc/cron.hourly
A directory containing shell scripts to be executed once per hour; see Cron Jobs.

/etc/cron.monthly
A directory containing shell scripts to be executed once per month; see Cron Jobs.

/etc/cron.weekly
A directory containing shell scripts to be executed once a week; see Cron Jobs.

/etc/init.d
A directory containing system initialization scripts; see Installation and Removal of Init Scripts.

/etc/profile.d
A directory containing shell scripts. Script names should follow the same conven-
tions as specified for cron jobs (see Cron Jobs, but should have the suffix .sh. The behavior is unspecified if a script is installed in this directory that does not have the suffix .sh.

The *sh utility shall read and execute commands in its current execution environment from all the shell scripts in this directory that have the suffix .sh when invoked as an interactive login shell, or if the -l (the letter ell) is specified (see Shell Invocation).

**Future Directions:** These directories are required at this version of the LSB since there is not yet an agreed method for abstracting the implementation so that applications need not be aware of these locations during installation.

### 18.2.1 File Naming Conventions

Conforming implementations and applications installing files into any of the above locations under /etc may only use filenames from the following managed namespaces:

- **Assigned names.** Such names must be chosen from the character set [a-z0-9]. In order to avoid conflicts these names shall be registered. This specification establishes a registry of provider, package and script names which is maintained at the Linux Assigned Names and Numbers Authority (LANANA). See www.lanana.org (http://www.lanana.org) to register names or look up already registered names.

  **Note:** Commonly used names should be registered to avoid conflicts and promote name reuse across distributions. Project developers are encouraged to reserve names with the LANANA as early as possible as registration is on a first-come, first-served basis.

- **Hierarchical names.** Script names in this category take the form: <hier1>-<hier2>-...-<name>, where name is taken from the character set [a-z0-9], and where there may be one or more <hier-n> components. <hier1> may either be an LSB provider name registered with the LANANA, or it may be a domain name registered to the provider in the DNS system, containing at least one "." (e.g. "debi-an.org", "staroffice.sun.com"). The LSB provider name registered with the LANANA shall only consist of the ASCII characters [a-z0-9].

  **Note:** As this specification cannot enforce rules for applications which do not choose to conform to it, conforming applications need to be aware that the managed namespaces may have been polluted with unregistered filenames and should check for namespace collisions and take appropriate steps if they occur.

  In general, if a package or system script is likely to be used on multiple systems, the package developers or the distribution should register the name through the LANANA, and distributions should strive to use the same name whenever possible. For applications which may not be essential or may not be commonly installed, the hierarchical namespace may be more appropriate. An advantage to the hierarchical namespace is that there is no need to consult with the LANANA before using a specific name.

- **Reserved names.** Names that begin with the character '_.' are reserved for distribution use only. Names in this form should be used for essential system packages only.

  **Note:** As this specification cannot enforce rules for applications which do not choose to conform to it, conforming applications need to be aware that the managed namespaces may have been polluted with unregistered filenames and should check for namespace collisions and take appropriate steps if they occur.

In general, if a package or system script is likely to be used on multiple systems, the package developers or the distribution should register the name through the LANANA, and distributions should strive to use the same name whenever possible. For applications which may not be essential or may not be commonly installed, the hierarchical namespace may be more appropriate. An advantage to the hierarchical namespace is that there is no need to consult with the LANANA before using a specific name.

Short names are highly desirable, since system administrators may wish to manually start and stop services. Given this, they should be standardized on a per-package basis. This is the rationale behind having the LANANA organization assign these names. The LANANA may be called upon to handle other namespace issues, such as package/prerequisites naming.

### 18.3 User Accounting Databases

The Filesystem Hierarchy Standard specifies two optional locations for user accounting databases used by the getutent(), getutent_r(), getutxent(), getutxid(), getutxline(), and pututxline() functions. These are /var/run/utmp and /var/run/wtmp.

The LSB does not specify the format or structure of these files, or even if they are files
at all. They should be used only as "magic cookies" to the utmpnam() function.

18.4 Path For System Administration Utilities

Certain utilities used for system administration (and other privileged commands) may be stored in /sbin, /usr/sbin, and /usr/local/sbin. Applications requiring to use commands identified as system administration utilities should add these directories to their PATH. By default, as described in POSIX_1003.1-2008 (ISO/IEC 9945-2009), standard utilities shall be found on the PATH returned by getconf PATH (or command -p getconf PATH to be guaranteed to invoke the correct version of getconf).
19 Additional Recommendations

19.1 Recommendations for applications on ownership and permissions

19.1.1 Directory Write Permissions

The application should not depend on having directory write permission in any directory except /tmp, /var/tmp, and the invoking user's home directory.

In addition, the application may store variable data in /var/opt/package, (where package is the name of the application package), if such a directory is created with appropriate permissions during the package installation.

For these directories the application should be able to work with directory write permissions restricted by the S_ISVTXT bit, implementing the restricted deletion mode as described for the XSI option for POSIX 1003.1-2008 (ISO/IEC 9945-2009).

19.1.2 File Write Permissions

The application should not depend on file write permission to any file that it does not itself create.

19.1.3 File Read and execute Permissions

The application should not depend on having read permission to every file and directory.

19.1.4 SUID and SGID Permissions

The application should not depend on the set user ID or set group ID (the S_ISUID or S_ISGID permission bits) permissions of a file not packaged with the application. Instead, the distribution is responsible for assuming that all system commands have the required permissions and work correctly.

**Rationale:** In order to implement common security policies it is strongly advisable for applications to use the minimum set of security attributes necessary for correct operation. Applications that require substantial appropriate privilege are likely to cause problems with such security policies.

19.1.5 Privileged users

In general, applications should not depend on running as a privileged user. This specification uses the term "appropriate privilege" throughout to identify operations that cannot be achieved without some special granting of additional privilege.

Applications that have a reason to run with appropriate privilege should outline this reason clearly in their documentation. Users of the application should be informed, that "this application demands security privileges, which could interfere with system security".

The application should not contain binary-only software that requires being run with appropriate privilege, as this makes security auditing harder or even impossible.

19.1.6 Changing permissions

The application shall not change permissions of files and directories that do not belong to its own package. Should an application require that certain files and directories not directly belonging to the package have a particular ownership, the application shall document this requirement, and may fail during installation if the permissions on these files...
is inappropriate.

19.1.7 Removable Media (Cdrom, Floppy, etc.)

Applications that expect to be runnable from removable media should not depend on logging in as a privileged user, and should be prepared to deal with a restrictive environment. Examples of such restrictions could be default mount options that disable set-user/group-ID attributes, disabling block or character-special files on the medium, or remapping the user and group IDs of files away from any privileged value.

Rationale: System vendors and local system administrators want to run applications from removable media, but want the possibility to control what the application can do.

19.1.8 Installable applications

Where the installation of an application needs additional privileges, it must clearly document all files and system databases that are modified outside of those in /opt/pkg-name, /etc/opt/pkg-name, and /var/opt/pkg-name, other than those that may be updated by system logging or auditing activities.

Without this, the local system administrator would have to blindly trust a piece of software, particularly with respect to its security.
20 Additional Behaviors

20.1 Mandatory Optional Behaviors

This section specifies behaviors in which there is optional behavior in one of the standards on which this specification relies, and where this specification requires a specific behavior.

Note: This specification does not require the kernel to be Linux; the set of mandated options reflects current existing practice, but may be modified in future releases.

LSB conforming implementations shall support the following options defined within the POSIX 1003.1-2008 (ISO/IEC 9945-2009):

- _POSIX_FSYNC
- _POSIX_MAPPED_FILES
- _POSIX_MEMLOCK
- _POSIX_MEMLOCK_RANGE
- _POSIX_MEMORY_PROTECTION
- _POSIX_PRIORITY_SCHEDULING
- _POSIX_REALTIME_SIGNALS
- _POSIX_THREAD_ATTR_STACKADDR
- _POSIX_THREAD_ATTR_STACKSIZE
- _POSIX_THREAD_PROCESS_SHARED
- _POSIX_THREAD_SAFE_FUNCTIONS
- _POSIX_THREADS

The opendir() function shall consume a file descriptor in the same fashion as open(), and therefore may fail with EMFILE or ENFILE.

The START and STOP termios characters shall be changeable, as described as optional behavior in the “General Terminal Interface” section of the POSIX 1003.1-2008 (ISO/IEC 9945-2009).

The access() function shall fail with errno set to EINVAL if the amode argument contains bits other than those set by the bitwise inclusive OR of R_OK, W_OK, X_OK and F_OK.

The link() function shall require access to the existing file in order to succeed, as described as optional behavior in the POSIX 1003.1-2008 (ISO/IEC 9945-2009).

Calling unlink() on a directory shall fail. Calling link() specifying a directory as the first argument shall fail. See also unlink.

Note: Linux allows rename() on a directory without having write access, but this specification does not require this behavior.

20.1.1 Special Requirements

LSB conforming systems shall enforce certain special additional restrictions above and beyond those required by POSIX 1003.1-2008 (ISO/IEC 9945-2009).

Note: These additional restrictions are required in order to support the testing and certification programs associated with the LSB. In each case, these are values that defined macros must not have; conforming applications that use these values shall trigger a failure in the interface that is otherwise described as a "may fail".

The fcntl() function shall treat the "cmd" value -1 as invalid.

The whence value -1 shall be an invalid value for the lseek(), fseek() and fcntl() functions.

The value -5 shall be an invalid signal number.
If the sigaddset() or sigdelset() functions are passed an invalid signal number, they shall return with EINVAL. Implementations are only required to enforce this requirement for signal numbers which are specified to be invalid by this specification (such as the -5 mentioned above).

The mode value -1 to the access() function shall be treated as invalid.

A value of -1 shall be an invalid "-_PC_..." value for pathconf().

A value of -1 shall be an invalid "-_SC_..." value for sysconf().

The nl_item value -1 shall be invalid for nl_langinfo().

The value -1 shall be an invalid "-_CS_..." value for confstr().

The value "a" shall be an invalid mode argument to popen().

The fcntl() function shall fail and set errno to EDEADLK if the cmd argument is F_SETLKW, and the lock is blocked by a lock from another process already blocked by the current process.

The opendir() function shall consume a file descriptor; the readdir() function shall fail and set errno to EBADF if the underlying file descriptor is closed.

The link() function shall not work across file systems, and shall fail and set errno to EXDEV as described as optional behavior in POSIX 1003.1-2008 (ISO/IEC 9945-2009).

20.2 Optional Mandatory Behaviors

This section specifies behaviors that are mandatory in one of the standards on which this specification relies, but which are optional in this specification.

POSIX 1003.1-2008 (ISO/IEC 9945-2009) describes the behavior of the file access time, available as the st_atime field of the stat and stat64 structures. An LSB conforming implementation need not update this information every time a file is accessed.

Note: A subsequent edition of the POSIX standard no longer mandates updating of st_atime but the older edition is still the guiding standard for this specification, thus this exception is needed.

20.3 Executable Scripts

An executable script is an executable file of which the first two characters are #! as defined in the portable character set. In POSIX 1003.1-2008 (ISO/IEC 9945-2009), this construct is undefined, but reserved for implementations which wish to provide this functionality. LSB conforming implementations shall support executable scripts.

A successful call to a function of the exec family with an executable script as the first parameter shall result in a new process, where the process image started is that of the interpreter. The path name of the interpreter follows the #! characters.

If the executable script has a first line

#! interpreter [arg]

then interpreter shall be called with an argument array consisting of an unspecified zeroth argument, followed by arg (if present), followed by a path name for the script, followed by the arguments following the zeroth argument in the exec call of the script.

The interpreter shall not perform any operations on the first line of an executable script. The first line of the executable script shall meet all of the following criteria otherwise the results are unspecified:

1. Is of one of the forms:

   #! interpreter
LSB Core - Generic 5.0

`#!/ interpreter
#!/ interpreter arg
#!/ interpreter arg`

2. The `interpreter` argument is an absolute pathname of an executable file other than an executable script.

3. Neither the `interpreter` argument nor the `arg` argument, if present, contain any quoting characters.

4. Neither the `interpreter` argument nor the `arg` argument, if present, contain any whitespace characters.

5. The length of the entire line is no longer than 80 bytes.

If the interpreter is required by this specification to be in a specific named directory, a conforming application must use that path for `interpreter`, as implementations are not prohibited from having other, possibly non-conforming, versions of the same interpreter installed on the system. If the interpreter is a required command in this specification, but does not have a required path, the application should take special measures to insure the appropriate version is selected. If the interpreter is not a required command in this specification, the application must make appropriate provisions that the interpreter is available at the appropriate path.

**Note:** In case the path is not specified, it is recommended that an installation script for executable scripts use the standard PATH returned by a call to the `getconf` command with the argument `PATH`, combined with the `command` command to determine the location of a standard command.

For example to determine the location of the standard `awk` command:

```
PATH=`getconf PATH` command -v awk
```

The installation script should ensure that the returned pathname is an absolute pathname prior to use, since a shell builtin might be returned for some utilities.

Use of the common form `#!/usr/bin/env interpreter` is not recommended as the PATH will be unknown at execution time and an alternative version of `interpreter` might be selected.
21 Localization

21.1 Introduction

In order to install a message catalog, the installation procedure shall supply the message catalog in a format readable by the **msgfmt** command, which shall be invoked to compile the message catalog into an appropriate binary format on the target system.

**Rationale:** The original intent was to allow an application to contain the binary GNU MO format files. However, the format of these files is not officially stable, hence it is necessary to compile these catalogs on the target system. These binary catalogs may differ from architecture to architecture as well.

The resulting binary message catalog shall be located in the package's private area under `/opt`, and the application may use `bindtextdomain()` to specify this location.

Implementations shall support the POSIX and C locales as specified in **POSIX 1003.1-2008 (ISO/IEC 9945-2009)**. Other locales may be supported.

Implementations may define additional locale categories not defined by that standard.

**Note:** Implementations choosing additional locale categories should be aware of **ISO/IEC TR14652** and are advised not to choose names that conflict with that specification. If implementations provide locale categories whose names are part of the FDCC set of **ISO/IEC TR14652**, they should behave as defined by that specification.

21.2 Regular Expressions

Utilities that process regular expressions shall support Basic Regular Expressions and Extended Regular Expressions as specified in **POSIX 1003.1-2008 (ISO/IEC 9945-2009)**, with the following exceptions:

- Range expression (such as `[a-z]`) can be based on code point order instead of collating element order.
- Equivalence class expression (such as `=[a]=`) and multi-character collating element expression (such as `[ .ch. ]`) are optional.
- Handling of a multi-character collating element is optional.

This affects at least the following utilities:

- `awk` (see `awk`)
- `grep` (see `grep`) (including `egrep`, see `egrep`)
- `sed` (see `sed`)

It also affects the behavior of interfaces in the base libraries, including at least

- `regexec()` (see `regexec`)

21.3 Pattern Matching Notation

Utilities that perform filename pattern matching (also known as Filename Globbing) shall do it as specified in **POSIX 1003.1-2008 (ISO/IEC 9945-2009)**, Pattern Matching Notation, with the following exceptions:

- Pattern bracket expressions (such as `[a-z]`) can be based on code point order instead of collating element order.
- Equivalence class expression (such as `=a=`) and multi-character collating element expression (such as `[ .ch. ]`) are optional.
- Handling of a multi-character collating element is optional.

This affects at least the following utilities: `cpio` (cpio), `find` and `tar` (tar).
VIII System Initialization
22 System Initialization

22.1 Cron Jobs

In addition to the individual user crontab files specified by POSIX 1003.1-2008 (ISO/IEC 9945-2009), which are located in /var/spool/cron as specified by the Filesystem Hierarchy Standard (FHS), the process that executes scheduled commands shall also process the following additional crontab files, which are in a different format (see below). /etc/crontab, /etc/cron.d/*. The installation of a package shall not modify the crontab file /etc/crontab, and shall not directly modify the user crontab files in /var/spool/cron/crontabs. but may use the crontab command to modify the latter.

If a package wishes to install a job that has to be executed periodically, it shall place an executable cron script in one of the following directories:
/etc/cron.hourly
/etc/cron.daily
/etc/cron.weekly
/etc/cron.monthly

As these directory names suggest, the files within them are executed on a hourly, daily, weekly, or monthly basis, respectively, under the control of an entry in one of the system crontab files, at an unspecified time of day. See below for the rules concerning the names of cron scripts.

Note: It is recommended that cron scripts installed in any of these directories be script files rather than compiled binaries so that they may be modified by the local system administrator. Conforming applications may only install cron scripts which use an interpreter required by this specification or provided by this or another conforming application.

This specification does not define the concept of a package upgrade. Implementations may do different things when packages are upgraded, including not replacing a cron script if it marked as a configuration file, particularly if the cron script appears to have been modified since installation. In some circumstances, the cron script may not be removed when the package is uninstalled. Applications should design their installation procedure and cron scripts to be robust in the face of such behavior. In particular, cron scripts should not fail obscurely if run in unexpected circumstances. Testing for the existence of application binaries before executing them is suggested.

If a certain task has to be executed at other than the predefined frequencies, the package shall install a file /etc/cron.d/cron-name. The file shall have the same format as that described for the crontab command in POSIX 1003.1-2008 (ISO/IEC 9945-2009), except that there shall be an additional field, username, before the name of the command to execute. For completeness, the seven fields shall be:

1. Minute [0,59]
2. Hour [0,23]
3. Day of the month [1,31]
4. Month of the year [1,12]
5. Day of the week [0,6] (with 0=Sunday)
6. Username
7. command [args ...]

This file shall be processed by the system automatically, with the named command being run at the specified time, as the specified username.

Applications installing files in these directories shall use the LSB naming conventions (see File Naming Conventions).
22.2 Init Script Actions

Conforming applications which need to execute commands on changes to the system run level (including boot and shutdown), may install one or more init scripts. Init scripts provided by conforming applications shall accept a single argument which selects the action:

- **start**: start the service
- **stop**: stop the service
- **restart**: stop and restart the service if the service is already running, otherwise start the service
- **try-restart**: restart the service if the service is already running
- **reload**: cause the configuration of the service to be reloaded without actually stopping and restarting the service
- **force-reload**: cause the configuration to be reloaded if the service supports this, otherwise restart the service if it is running
- **status**: print the current status of the service

The **start**, **stop**, **restart**, **force-reload**, and **status** actions shall be supported by all init scripts; the **reload** and the **try-restart** actions are optional. Other init-script actions may be defined by the init script.

Init scripts shall ensure that they will behave sensibly if invoked with **start** when the service is already running, or with **stop** when not running, and that they do not kill similarly-named user processes. The best way to achieve this is to use the init-script functions provided by /lib/lsb/init-functions (see **Init Script Functions**).

If a service reloads its configuration automatically (as in the case of cron, for example), the **reload** action of the init script shall behave as if the configuration was reloaded successfully. The **restart**, **try-restart**, **reload** and **force-reload** actions may be atomic; that is if a service is known not to be operational after a restart or reload, the script may return an error without any further action.

**Note**: This specification does not define the concept of a package upgrade. Implementations may do different things when packages are upgraded, including not replacing an init script if it is marked as a configuration file, particularly if the file appears to have been modified since installation. In some circumstances, the init script may not be removed when the package is uninstalled. Applications should design their installation procedure and init scripts to be robust in the face of such behavior. In particular, init scripts should not fail obscurely if run in unexpected circumstances. Testing for the existence of application binaries before executing them is suggested.

If the **status** action is requested, the init script will return the following exit status codes.

- **0**: program is running or service is OK
- **1**: program is dead and /var/run pid file exists
- **2**: program is dead and /var/lock lock file exists
- **3**: program is not running
- **4**: program or service status is unknown
- **5-99**: reserved for future LSB use
- **100-149**: reserved for distribution use
- **150-199**: reserved for application use
- **200-254**: reserved

For all other init-script actions, the init script shall return an exit status of zero if the action was successful. Otherwise, the exit status shall be non-zero, as defined below. In addition to straightforward success, the following situations are also to be considered successful:
LSB Core - Generic 5.0

- restarting a service (instead of reloading it) with the **force-reload** argument
- running **start** on a service already running
- running **stop** on a service already stopped or not running
- running **restart** on a service already stopped or not running
- running **try-restart** on a service already stopped or not running

In case of an error while processing any init-script action except for **status**, the init script shall print an error message and exit with a non-zero status code:

1. generic or unspecified error (current practice)
2. invalid or excess argument(s)
3. unimplemented feature (for example, "reload")
4. user had insufficient privilege
5. program is not installed
6. program is not configured
7. program is not running
8-99 reserved for future LSB use
100-149 reserved for distribution use
150-199 reserved for application use
200-254 reserved

Error and status messages should be printed with the logging functions (see **Init Script Functions**) log_success_msg(), log_failure_msg() and log_warning_msg(). Scripts may write to standard error or standard output, but implementations need not present text written to standard error/output to the user or do anything else with it.

**Note:** Since init scripts may be run manually by a system administrator with non-standard environment variable values for PATH, USER, LOGNAME, etc., init scripts should not depend on the values of these environment variables. They should set them to some known/default values if they are needed.

### 22.3 Comment Conventions for Init Scripts

Conforming applications may install one or more init scripts. These init scripts must be activated by invoking the **install_initd** command. Prior to package removal, the changes applied by **install_initd** must be undone by invoking **remove_initd**. See **Installation and Removal of Init Scripts** for more details.

**install_initd** and **remove_initd** determine actions to take by decoding a specially formatted block of lines in the script. This block shall be delimited by the lines

```plaintext
### BEGIN INIT INFO
### END INIT INFO
```

The delimiter lines may contain trailing whitespace, which shall be ignored. All lines inside the block shall begin with a hash character ' #' in the first column, so the shell interprets them as comment lines which do not affect operation of the script. The lines shall be of the form:

```plaintext
# {keyword}: arg1 [arg2...]
```

with exactly one space character between the ' #' and the keyword, with a single exception. In lines following a line containing the **Description** keyword, and until the next keyword or block ending delimiter is seen, a line where the ' #' is followed by more than one space or a tab character shall be treated as a continuation of the previous line.

The information extracted from the block is used by the installation tool or the init-script system to assure that init scripts are run in the correct order. It is unspecified whether the information is evaluated only when **install_initd** runs, when the init scripts are ex-
executed, or both. The information extracted includes run levels, defined in Run Levels, and boot facilities, defined in Facility Names.

The following keywords, with their arguments, are defined:

**Provides:** boot_facility_1 [boot_facility_2...]

boot facilities provided by this init script. When an init script is run with a **start** argument, the boot facility or facilities specified by the **Provides** keyword shall be deemed present and hence init scripts which require those boot facilities should be started later. When an init script is run with a **stop** argument, the boot facilities specified by the **Provides** keyword are deemed no longer present.

**Required-Start:** boot_facility_1 [boot_facility_2...]

facilities which must be available during startup of this service. The init-script system should ensure init scripts which provide the **Required-Start** facilities are started before starting this script.

**Required-Stop:** boot_facility_1 [boot_facility_2...]

facilities which must be available during the shutdown of this service. The init-script system should avoid stopping init scripts which provide the **Required-Stop** facilities until this script is stopped.

**Should-Start:** boot_facility_1 [boot_facility_2...]

facilities which, if present, should be available during startup of this service. This allows for weak dependencies which do not cause the service to fail if a facility is not available. The service may provide reduced functionality in this situation. Conforming applications should not rely on the existence of this feature.

**Should-Stop:** boot_facility_1 [boot_facility_2...]

facilities which should be available during shutdown of this service.

**Default-Start:** run_level_1 [run_level_2...]

**Default-Stop:** run_level_1 [run_level_2...]

which run levels should by default run the init script with a **start** (**stop**) argument to start (stop) the services controlled by the init script.

For example, if a service should run in runlevels 3, 4, and 5 only, specify "Default-Start: 3 4 5" and "Default-Stop: 0 1 2 6".

**Short-Description:** short_description

provide a brief description of the actions of the init script. Limited to a single line of text.

**Description:** multiline_description

provide a more complete description of the actions of the init script. May span multiple lines. In a multiline description, each continuation line shall begin with a ‘#’ followed by tab character or a ‘#’ followed by at least two space characters. The multiline description is terminated by the first line that does not match this criteria.

Additional keywords may be defined in future versions of this specification. Also, implementations may define local extensions by using the prefix **X-implementor**. For example, X-RedHat-foobardecl, or X-Debian-xyzzydecl.

Example:

```bash
### BEGIN INIT INFO
# Provides: lsb-ourdb
# Required-Start: $local_fs $network $remote_fs
# Required-Stop: $local_fs $network $remote_fs
```
The comment conventions described in this section are only required for init scripts installed by conforming applications. Conforming runtime implementations are not required to use this scheme in their system provided init scripts.

Note: This specification does not require, but is designed to allow, the development of a system which runs init scripts in parallel. Hence, enforced-serialization of scripts is avoided unless it is explicitly necessary.

### 22.4 Installation and Removal of Init Scripts

Conforming applications may install one or more initialization scripts (or init scripts). An init script shall be installed in `/etc/init.d` (which may be a symbolic link to another location), by the package installer.

During the installer's post-install processing phase the program `/usr/lib/lsb/install_initd` must be called to activate the init script. Activation consists of arranging for the init script to be called in the correct order on system run-level changes (including system boot and shutdown), based on dependencies supplied in the init script (see Comment Conventions for Init Scripts). The `install_initd` command should be thought of as a wrapper which hides the implementation details; how any given implementation arranges for the init script to be called at the appropriate time is not specified.

Example: if an init script specified "Default-Start: 3 4 5" and "Default-Stop: 0 1 2 6", `install_initd` might create "start" symbolic links with names starting with 'S' in `/etc/rc3.d`, `/etc/rc4.d` and `/etc/rc5.d` and "stop" symbolic links with names starting with 'K' in `/etc/rc0.d`, `/etc/rc1.d`, `/etc/rc2.d` and `/etc/rc6.d`. Such a scheme would be similar to the System V Init mechanism, but is by no means the only way this specification could be implemented.

The `install_initd` command takes a single argument, the full pathname of the installed init script. The init script must already be installed in `/etc/init.d`. The `install_initd` command will not copy it there, only activate it once it has been installed. For example:

```
/usr/lib/lsb/install_initd /etc/init.d/example.com-coffeed
```

The `install_initd` command shall return an exit status of zero if the init-script activation was successful or if the init script was already activated. If the dependencies in the init script (see Comment Conventions for Init Scripts) cannot be met, an exit status of one shall be returned and the init script shall not be activated.

When a software package is removed, `/usr/lib/lsb/remove_initd` must be called to deactivate the init script. This must occur before the init script itself is removed, as the dependency information in the script may be required for successful completion. Thus the installer's pre-remove processing phase must call `remove_initd`, and pass the full pathname of the installed init script. The package installer is still responsible for removing the init script. For example:

```
/usr/lib/lsb/remove_initd /etc/init.d/example.com-coffeed
```

The `remove_initd` program shall return an exit status of zero if the init script has been successfully deactivated or if the init script is not activated. If another init script which depends on a boot facility provided by this init script is activated, an exit status of one shall be returned and the init script shall remain activated. The installer must fail on such an exit code so it does not subsequently remove the init script.

Note: This specification does not describe a mechanism for the system administrator to
manipulate the run levels at which an init script is started or stopped. There is no assurance that modifying the comment block for this purpose will have the desired effect.

22.5 Run Levels

The following run levels are specified for use by the Default-Start and Default-Stop actions defined in Comment Conventions for Init Scripts as hints to the install_initd command. Conforming implementations are not required to provide these exact run levels or give them the meanings described here, and may map any level described here to a different level which provides the equivalent functionality. Applications may not depend on specific run-level numbers.

<table>
<thead>
<tr>
<th>Run Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>halt</td>
</tr>
<tr>
<td>1</td>
<td>single user mode</td>
</tr>
<tr>
<td>2</td>
<td>multiuser with no network services exported</td>
</tr>
<tr>
<td>3</td>
<td>normal/full multiuser</td>
</tr>
<tr>
<td>4</td>
<td>reserved for local use, default is normal/full multiuser</td>
</tr>
<tr>
<td>5</td>
<td>multiuser with a display manager or equivalent</td>
</tr>
<tr>
<td>6</td>
<td>reboot</td>
</tr>
</tbody>
</table>

Note: These run levels were chosen as reflecting the most frequent existing practice, and in the absence of other considerations, implementors are strongly encouraged to follow this convention to provide consistency for system administrators who need to work with multiple distributions.

22.6 Facility Names

Boot facilities are used to indicate dependencies in initialization scripts, as defined in Comment Conventions for Init Scripts. Facility names are assigned to scripts by the Provides: keyword. Facility names that begin with a dollar sign ('$') are reserved system facility names.

Note: Facility names are only recognized in the context of the init script comment block and are not available in the body of the init script. In particular, the use of the leading 'S' character does not imply system facility names are subject to shell variable expansion, since they appear inside comments.

Conforming applications shall not provide facilities that begin with a dollar sign. Implementations shall provide the following facility names:

$local_fs
all local file systems are mounted

$network
basic networking support is available. Example: a server program could listen on a socket.

$named
IP name-to-address translation, using the interfaces described in this specification, are available to the level the system normally provides them. Example: if a DNS query daemon normally provides this facility, then that daemon has been started.

$portmap
daemons providing SunRPC/ONCRPC portmapping service as defined in RFC 1833: Binding Protocols for ONC RPC Version 2 (if present) are running.
$remote_fs

all remote file systems are available. In some configurations, file systems such as /usr may be remote. Many applications that require $local_fs will probably also require $remote_fs.

$syslog

system logger is operational.

$time

the system time has been set, for example by using a network-based time program such as ntp or rdate, or via the hardware Real Time Clock.

Other (non-system) facilities may be defined by other conforming applications. These facilities shall be named using the same conventions defined for naming init scripts (see Script Names). Commonly, the facility provided by a conforming init script will have the same name as the name assigned to the init script.

22.7 Script Names

Since init scripts live in a single directory, they must share a single namespace. To avoid conflicts, applications installing files in this directories shall use the LSB naming conventions (see File Naming Conventions).

22.8 Init Script Functions

Each conforming init script shall execute the commands in the file /lib/lsb/init-functions in the current environment (see shell special built-in command dot). This file shall cause the following shell script commands to be defined in an unspecified manner.

Note: This can be done either by adding a directory to the PATH variable which defines these commands, or by defining shell aliases or functions.

Although the commands made available via this mechanism need not be conforming applications in their own right, applications that use them should only depend on features described in this specification.

Conforming scripts shall not specify the "exit on error" option (i.e. set -e) when sourcing this file, or calling any of the commands thus made available.

The start_daemon, killproc and pidofproc functions shall use the following algorithm for determining the status and the process identifiers of the specified program.

1. If the -p pidfile option is specified, and the named pidfile exists, a single line at the start of the pidfile shall be read. If this line contains one or more numeric values, separated by spaces, these values shall be used. If the -p pidfile option is specified and the named pidfile does not exist, the functions shall assume that the daemon is not running.

2. Otherwise, /var/run/basename.pid shall be read in a similar fashion. If this contains one or more numeric values on the first line, these values shall be used. Optionally, implementations may use unspecified additional methods to locate the process identifiers required.

The method used to determine the status is implementation defined, but should allow for non-binary programs.

Note: Commonly used methods check either for the existence of the /proc/pid directory or use /proc/pid/exe and /proc/pid/cmdline. Relying only on /proc/pid/exe is discouraged since this specification does not specify the existence of, or semantics for, /proc. Additionally, using /proc/pid/exe may result in a not-running status for daemons that are written in a script language.
Conforming implementations may use other mechanisms besides those based on pidfiles, unless the \texttt{-p pidfile} option has been used. Conforming applications should not rely on such mechanisms and should always use a pidfile. When a program is stopped, it should delete its pidfile. Multiple process identifiers shall be separated by a single space in the pidfile and in the output of \texttt{pidofproc}.

\texttt{start\_daemon [-f] [-n nicelevel] [-p pidfile] pathname [args...]} runs the specified program as a daemon. The \texttt{start\_daemon} function shall check if the program is already running using the algorithm given above. If so, it shall not start another copy of the daemon unless the \texttt{-f} option is given. The \texttt{-n} option specifies a nice level. See \texttt{nice}. \texttt{start\_daemon} shall return the LSB defined exit status codes. It shall return 0 if the program has been successfully started or is running and not 0 otherwise.

\texttt{killproc [-p pidfile] pathname [signal]} The \texttt{killproc} function shall stop the specified program. The program is found using the algorithm given above. If a signal is specified, using the \texttt{-signal\_name} or \texttt{-signal\_number} syntaxes as specified by the \texttt{kill} command, the program is sent that signal. Otherwise, a SIGTERM followed by a SIGKILL after an unspecified number of seconds shall be sent. If a program has been terminated, the pidfile should be removed if the terminated process has not already done so. The \texttt{killproc} function shall return the LSB defined exit status codes. If called without a signal, it shall return 0 if the program has been stopped or is not running and not 0 otherwise. If a signal is given, it shall return 0 only if the program is running.

\texttt{pidofproc [-p pidfile] pathname} The \texttt{pidofproc} function shall return one or more process identifiers for a particular daemon using the algorithm given above. Only process identifiers of running processes should be returned. Multiple process identifiers shall be separated by a single space.

\textbf{Note:} A process may exit between \texttt{pidofproc} discovering its identity and the caller of \texttt{pidofproc} being able to act on that identity. As a result, no test assertion can be made that the process identifiers returned by \texttt{pidofproc} shall be running processes.

The \texttt{pidofproc} function shall return the LSB defined exit status codes for “status”. It shall return 0 if the program is running and not 0 otherwise.

\texttt{log\_success\_msg message} The \texttt{log\_success\_msg} function shall cause the system to write a success message to an unspecified log file. The format of the message is unspecified. The \texttt{log\_success\_msg} function may also write a message to the standard output.

\textbf{Note:} The message should be relatively short; no more than 60 characters is highly desirable.

\texttt{log\_failure\_msg message} The \texttt{log\_failure\_msg} function shall cause the system to write a failure message to an unspecified log file. The format of the message is unspecified. The \texttt{log\_failure\_msg} function may also write a message to the standard output.

\textbf{Note:} The message should be relatively short; no more than 60 characters is highly desirable.

\texttt{log\_warning\_msg message} The \texttt{log\_warning\_msg} function shall cause the system to write a warning message to an unspecified log file. The format of the message is unspecified. The \texttt{log\_warning\_msg} function may also write a message to the standard output.
Note: The message should be relatively short; no more than 60 characters is highly desirable.
IX Users & Groups
23 Users & Groups

23.1 User and Group Database

The format of the User and Group databases is not specified. Programs may only read these databases using the provided API. Changes to these databases should be made using the provided commands.

23.2 User & Group Names

Table 23-1 describes required mnemonic user and group names. This specification makes no attempt to numerically assign user or group identity numbers, with the exception that both the User ID and Group ID for the user root shall be equal to 0.

Table 23-1 Required User & Group Names

<table>
<thead>
<tr>
<th>User</th>
<th>Group</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>root</td>
<td>root</td>
<td>Administrative user with all appropriate privileges</td>
</tr>
<tr>
<td>bin</td>
<td>bin</td>
<td>Legacy User ID/Group ID^a</td>
</tr>
<tr>
<td>daemon</td>
<td>daemon</td>
<td>Legacy User ID/Group ID^b</td>
</tr>
</tbody>
</table>

Notes:

a The bin User ID/Group ID is included for compatibility with legacy applications. New applications should no longer use the bin User ID/Group ID.

b The daemon User ID/Group ID was used as an unprivileged User ID/Group ID for daemons to execute under in order to limit their access to the system. Generally daemons should now run under individual User ID/Group IDs in order to further partition daemons from one another.

Table 23-2 is a table of optional mnemonic user and group names. This specification makes no attempt to numerically assign uid or gid numbers. If the username exists on a system, then they should be in the suggested corresponding group. These user and group names are for use by distributions, not by applications.

Table 23-2 Optional User & Group Names

<table>
<thead>
<tr>
<th>User</th>
<th>Group</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>adm</td>
<td>adm</td>
<td>Administrative special privileges</td>
</tr>
<tr>
<td>lp</td>
<td>lp</td>
<td>Printer special privileges</td>
</tr>
<tr>
<td>sync</td>
<td>sync</td>
<td>Login to sync the system</td>
</tr>
<tr>
<td>shutdown</td>
<td>shutdown</td>
<td>Login to shutdown the system</td>
</tr>
<tr>
<td>halt</td>
<td>halt</td>
<td>Login to halt the system</td>
</tr>
<tr>
<td>mail</td>
<td>mail</td>
<td>Mail special privileges</td>
</tr>
<tr>
<td>news</td>
<td>news</td>
<td>News special privileges</td>
</tr>
<tr>
<td>uucp</td>
<td>uucp</td>
<td>UUCP special privileges</td>
</tr>
<tr>
<td>operator</td>
<td>root</td>
<td>Operator special privileges</td>
</tr>
<tr>
<td>man</td>
<td>man</td>
<td>Man special privileges</td>
</tr>
<tr>
<td>nobody</td>
<td>nobody</td>
<td>Used by NFS</td>
</tr>
</tbody>
</table>
Only a minimum working set of "user names" and their corresponding "user groups" are required. Applications cannot assume non system user or group names will be defined.

Applications cannot assume any policy for the default file creation mask (umask) or the default directory permissions a user may have. Applications should enforce user only file permissions on private files such as mailboxes. The location of the users home directory is also not defined by policy other than the recommendations of the Filesystem Hierarchy Standard and should be obtained by the getpwnam(), getpwnam_r(), getpwent(), getpwuid(), and getpwuid_r() functions.

23.3 User ID Ranges

The system User IDs from 0 to 99 should be statically allocated by the system, and shall not be created by applications.

The system User IDs from 100 to 499 should be reserved for dynamic allocation by system administrators and post install scripts using useradd.

23.4 Rationale

The purpose of specifying optional users and groups is to reduce the potential for name conflicts between applications and distributions.
X Network Security Services
24 Libraries

24.1 Interfaces for libnspr4

Table 24-1 defines the library name and shared object name for the libnspr4 library

<table>
<thead>
<tr>
<th>Library:</th>
<th>libnspr4</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libnspr4.so</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications: [NSPR] NSPR Reference

24.1.1 Netscape Portable Runtime

24.1.1.1 Interfaces for Netscape Portable Runtime

An LSB conforming implementation shall provide the generic functions for Netscape Portable Runtime specified in Table 24-2, with the full mandatory functionality as described in the referenced underlying specification.

Table 24-2 libnspr4 - Netscape Portable Runtime Function Interfaces

<table>
<thead>
<tr>
<th>PR_Abort [NSPR]</th>
<th>PR_Accept [NSPR]</th>
<th>PR_AtomDecrementIncrement [NSPR]</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR_Bind [NSPR]</td>
<td>PR_Calloc [NSPR]</td>
<td>PR_Cleanup [NSPR]</td>
</tr>
<tr>
<td>PR_ConnectContinue [NSPR]</td>
<td>PR_ConvertIPv4AddrToIPv6 [NSPR]</td>
<td>PR_CreateIOLayerStub [NSPR]</td>
</tr>
<tr>
<td>PR_Free [NSPR]</td>
<td>PR_FreeAddrInfo [NSPR]</td>
<td>PR_GMTParameters [NSPR]</td>
</tr>
<tr>
<td>PR_GetAddrInfoByName [NSPR]</td>
<td>PR_GetCanonNameFromAddrInfo [NSPR]</td>
<td>PR_GetDefaultIOMethods [NSPR]</td>
</tr>
<tr>
<td>PR_ImportTCPSocket [NSPR]</td>
<td>PR_Init [NSPR]</td>
<td>PR_Initialize [NSPR]</td>
</tr>
<tr>
<td>PR_InitializeNetAddr [NSPR]</td>
<td>PR_IntervalToMicroseconds [NSPR]</td>
<td>PR_IntervalToMilliseconds [NSPR]</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
24.2 Data Definitions for libnspr4

This section defines global identifiers and their values that are associated with interfaces contained in libnspr4. 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.

### 24.2.1 nspr4/nspr.h

```c
#define nspr_h___
```

### 24.2.2 nspr4/plarena.h

```c
#define plarena_h___
```
typedef struct PLArenaPool {
    struct PLArena first;
    struct PLArena *current;
    PRUint32 arenasize;
    PRUword mask;
} PLArenaPool;
struct PLArena {
    struct PLArena *next;
    PRUword base;
    PRUword limit;
    PRUword avail;
};

24.2.3 nspr4/plhash.h

#define plhash_h___
typedef PRUint32 PLHashNumber;
typedef PRIntn("PLHashComparator) (const void *, const void *");
typedef struct PLHashAllocOps {
    void *("allocTable) (void *, PRSize);
    void (*freeTable) (void *, void *);
    struct PLHashEntry *("allocEntry) (void *, const void *");
    void (*freeEntry) (void *, struct PLHashEntry *, PRUintn);
} PLHashAllocOps;
typedef PLHashNumber("PLHashFunction) (const void *");
struct PLHashEntry {
    struct PLHashEntry *next;
    PLHashNumber keyHash;
    const void *key;
    void *value;
};
struct PLHashTable {
    struct PLHashEntry **buckets;
    PRUint32 nentries;
    PRUint32 shift;
    PLHashFunction keyHash;
    PLHashComparator keyCompare;
    PLHashComparator valueCompare;
    const PLHashAllocOps *allocOps;
    void *allocPriv;
};

24.2.4 nspr4/pratom.h

#define PR_ATOMIC_ADD(val)      PR_AtomicAdd(val)
#define PR_ATOMIC_DECREMENT(val)        PR_AtomicDecrement(val)
#define PR_ATOMIC_INCREMENT(val)        PR_AtomicIncrement(val)
#define PR_ATOMIC_SET(val)      PR_AtomicSet(val)

extern PRInt32 PR_AtomicAdd(PRInt32 * ptr, PRInt32 val);
extern PRInt32 PR_AtomicDecrement(PRInt32 * val);
extern PRInt32 PR_AtomicIncrement(PRInt32 * val);
extern PRInt32 PR_AtomicSet(PRInt32 * val, PRInt32 newval);

24.2.5 nspr4/prclist.h

#define prclist_h___
typedef struct PRCListStr {
    PRCList *next;
    PRCList *prev;
}
24.2.6 nspr4/prcvar.h

typedef struct PRCondVar PRCondVar;
extern void PR_DestroyCondVar(PRCondVar * cvar);
extern void PR_DestroyCondVar(PRCondVar * cvar);
extern PRCondVar *PR_NewCondVar(PRLock * lock);
extern PRStatus PR_NotifyAllCondVar(PRCondVar * cvar);
extern PRStatus PR_NotifyCondVar(PRCondVar * cvar);
extern PRStatus PR_WaitCondVar(PRCondVar * cvar, PRIntervalTime timeout);

24.2.7 nspr4/prerror.h

#define prerror_h___
typedef PRInt32 PRErrorCode;
extern PRErrorCode PR_GetError(void);
extern PRInt32 PR_GetErrorText(char *text);
extern PRInt32 PR_GetErrorTextLength(void);
extern PRInt32 PR_GetOSError(void);
extern void PR_SetError(PRErrorCode errorCode, PRInt32 oserr);
extern void PR_SetErrorText(PRIntn textLength, const char *text);

24.2.8 nspr4/prinit.h

#define prinit_h___
typedef PRIntn(*PRPrimordialFn) (PRIntn argc, char **argv);
typedef PRStatus(*PRCallOnceFN) (void);
typedef PRStatus(*PRCallOnceWithArgFN) (void *arg);
extern void PR_Abort(void);
extern PRStatus PR_Cleanup(void);
extern void PR_Init(PRThreadType type, PRThreadPriority priority,
PRUintn maxPTDs);
extern PRIntn PR_Initialize(PRPrimordialFn prmain, PRIntn argc,
char **argv, PRUintn maxPTDs);
extern PRBool PR_Initialized(void);
extern void PR_ProcessExit(PRIntn status);

24.2.9 nspr4/prinrval.h

#define prinrval_h
#define PR_INTERVAL_NO_WAIT     0UL
#define PR_INTERVAL_NO_TIMEOUT  0xffffffffUL
typedef PRUint32 PRIntervalTime;
extern PRIntervalTime PR_IntervalNow(void);
extern PRUint32 PR_IntervalToMicroseconds(PRIntervalTime ticks);
extern PRUint32 PR_IntervalToMilliseconds(PRIntervalTime ticks);
extern PRUint32 PR_IntervalToSeconds(PRIntervalTime ticks);
extern PRIntervalTime PR_MicrosecondsToInterval(PRUint32 micro);
extern PRIntervalTime PR_MillisecondsToInterval(PRUint32 milli);
extern PRIntervalTime PR_SecondsToInterval(PRUint32 seconds);
extern PRUint32 PR_TicksPerSecond(void);

24.2.10 nspr4/prio.h
#define prio_h___
#define PR_RDONLY       0x01
#define PR_WRONLY       0x02
#define PR_RDWR 0x04
#define PR_CREATE_FILE 0x08
#define PR_APPEND       0x10
#define PR_TRUNCATE     0x20
#define PR_SYNC 0x40
#define PR_EXCL 0x80

typedef enum PRDescType {
    PR_DESC_FILE = 1,
    PR_DESC_SOCKET_TCP = 2,
    PR_DESC_SOCKET_UDP = 3,
    PR_DESC_LAYERED = 4,
    PR_DESC_PIPE = 5
} PRDescType;

typedef struct PRIPv6Addr {
    union {
        PRUint8 _S6_u8[15];
        PRUint16 _S6_u16[7];
        PRUint32 _S6_u32[3];
        PRUint64 _S6_u64[1];
    } _S6_un;
} PRIPv6Addr;

typedef enum PRTransmitFileFlags {
    PR_TRANSMITFILE_KEEP_OPEN = 0,
    PR_TRANSMITFILE_CLOSE_SOCKET = 1
} PRTransmitFileFlags;

typedef struct PRLinger {
    PRBool polarity;
    PRIntervalTime linger;
} PRLinger;

typedef struct PRFilePrivate PRFilePrivate;

typedef struct PRFileDesc {
    const struct PRIOMethods *methods;
    PRFilePrivate *secret;
    PRFileDesc *lower;
    PRFileDesc *higher;
    void (*dtor) (PRFileDesc *);
    PRDescIdentity identity;
} PRFileDesc;

typedef union PRNetAddr {
    struct {
        PRUint16 family;
        char data[14];
    } raw;
    struct {
        PRUint16 family;
        PRUint16 port;
        PRUint32 ip;
        char pad[7];
    } inet;
    struct {
        PRUint16 family;
        PRUint16 port;
        PRUint32 flowinfo;
        PRIPv6Addr ip;
        PRUint32 scope_id;
    } ipv6;
    struct {
        PRUint16 family;
        char path[103];
    } local;
} PRNetAddr;

typedef struct PRMcastRequest {
union PRNetAddr mcaddr;
union PRNetAddr ifaddr;
} PRMcastRequest;
typedef struct PRIOVec {
    char *iov_base;
    int iov_len;
} PRIOVec;
typedef struct PRSocketOptionData {
    PRSockOption option;
    union {
        PRUintn ip_ttl;
        PRUintn mcast_ttl;
        PRUintn tos;
        PRBool non_blocking;
        PRBool reuse_addr;
        PRBool keep_alive;
        PRBool mcast_loopback;
        PRBool no_delay;
        PRBool broadcast;
        PRSize max_segment;
        PRSize recv_buffer_size;
        PRSize send_buffer_size;
        PRLinger linger;
        PRMcastRequest add_member;
        PRMcastRequest drop_member;
    union PRNetAddr mcast_if;
} value;
} PRSocketOptionData;
typedef PRStatus("PRSyncFN) (PRFileDesc * fd);
typedef PRStatus("PRListenFN) (PRFileDesc * fd, PRIntn how);
typedef enum PRSeekWhence {
    PR_SEEK_SET = 0,
    PR_SEEK_CUR = 1,
    PR_SEEK_END = 2
} PRSeekWhence;
typedef PRInt32("PRAcceptreadFN) (PRFileDesc * sd, PRFileDesc * *nd,
    PRNetAddr * *raddr, void *buf,
    PRInt32 amount, PRIntervalTime t);
typedef PRStatus("PRCloseFN) (PRFileDesc * fd);
typedef PRInt32("PRTransmitfileFN) (PRFileDesc * sd, PRFileDesc * fd,
    const void *headers, PRInt32 hlen,
    PRTransmitFileFlags flags,
    PRIntervalTime t);
typedef enum PRSockOption {
    PR_SockOpt_Nonblocking,
    PR_SockOpt_Linger = 1,
    PR_SockOpt_Reuseaddr = 2,
    PR_SockOpt_Keepalive = 3,
    PR_SockOpt_RecvBufferSize = 4,
    PR_SockOpt_SendBufferSize = 5,
    PR_SockOpt_IpTimeToLive = 6,
    PR_SockOpt_IpTypeOfService = 7,
    PR_SockOpt_AddMember = 8,
    PR_SockOpt_DropMember = 9,
    PR_SockOpt_McastInterface = 10,
    PR_SockOpt_McastTimeToLive = 11,
    PR_SockOpt_McastLoopback = 12,
    PR_SockOpt_NoDelay = 13,
    PR_SockOpt_MaxSegment = 14,
    PR_SockOpt_Broadcast = 15,
    PR_SockOpt_Last = 16
} PRSockOption;
typedef PRFileDesc *(*PRAcceptFN) (PRFileDesc * fd, PRNetAddr * addr, PRIntervalTime timeout);
typedef PRStatus(*PRConnectcontinueFN) (PRFileDesc * fd, PRInt16 out_flags);
typedef PRInt32(*PRReadFN) (PRFileDesc * fd, void *buf, PRInt32 amount);
typedef struct PRFileInfo64 {
  PRFileType type;
  PROffset64 size;
  PRTime creationTime;
  PRTime modifyTime;
} PRFileInfo64;
typedef PRStatus(*PRGetsockoptoptionFN) (PRFileDesc * fd, PRSocketOptionData * data);
typedef PRInt32(*PRSendtoFN) (PRFileDesc * fd, const void *buf, PRInt32 amount, PRIntn flags, const PRNetAddr * addr, PRIntervalTime timeout);
typedef PRStatus(*PRGetsocknameFN) (PRFileDesc * fd, PRNetAddr * addr);
typedef PRInt32(*PRSendFN) (PRFileDesc * fd, const void *buf, PRInt32 amount, PRIntn flags, PRIntervalTime timeout);
typedef  PROffset32(*PRSeekFN)  (PRFileDesc  *  fd,  PROffset32 offset, PRSeekWhence how);
typedef PRInt64(*PRAvailable64FN) (PRFileDesc * fd);
typedef PRInt32(*PRAvailableFN) (PRFileDesc * fd);
typedef struct PRFileInfo {
  PRFileType type;
  PROffset32 size;
  PRTime creationTime;
  PRTime modifyTime;
} PRFileInfo;
typedef  PROffset64(*PRSeek64FN)  (PRFileDesc  *  fd,  PROffset64 offset, PRSeekWhence how);
typedef PRStatus(*PRSetsockoptoptionFN) (PRFileDesc * fd, const PRSocketOptionData * data);
typedef PRInt32(*PRRecvFN) (PRFileDesc * fd, void *buf, PRInt32 amount, PRIntn flags, PRIntervalTime timeout);
typedef struct PRSendFileData {
  PRFileDesc *fd;
  PRUint32 file_offset;
  PRSize file_nbytes;
  const void *header;
  PRInt32 hlen;
  const void *trailer;
  PRInt32 tlen;
} PRSendFileData;
typedef PRIntn PRDescIdentity;
typedef PRStatus(*PRConnectFN) (PRFileDesc * fd, const PRNetAddr * addr, PRIntervalTime timeout);
typedef PRInt32(*PRSendfileFN) (PRFileDesc * networkSocket, PRSendFileData * sendData, PRTransmitFileFlags flags, PRIntervalTime timeout);
typedef PRInt32(*PRRecvfromFN) (PRFileDesc * fd, void *buf, PRInt32 amount, PRIntn flags, PRNetAddr * addr,
typedef struct PRPollDesc {
    PRFileDesc *fd;
    PRInt16 in_flags;
    PRInt16 out_flags;
} PRPollDesc;

typedef PRInt32(*PRWriteFN) (PRFileDesc * fd, const void *buf,
    PRInt32 amount);

typedef PRStatus(*PRFilename64FN) (PRFileDesc * fd, PRFileInfo64
    * info);

typedef PRStatus(*PRShutdownFN) (PRFileDesc * fd, PRIntn how);

typedef PRIntn(*PRReservedFN) (PRFileDesc * fd);

typedef PRStatus(*PRFileInfoFN) (PRFileDesc * fd, PRFileInfo *
    info);

typedef PRStatus(*PRWritevFN) (PRFileDesc * fd, const PRIOVec *
    iov,
    PRInt32 iov_size, PRIntervalTime
    timeout);

typedef enum PRFileType {
    PR_FILE_FILE = 1,
    PR_FILE_DIRECTORY = 2,
    PR_FILE_OTHER = 3
} PRFileType;

typedef PRStatus(*PRBindFN) (PRFileDesc * fd, const PRNetAddr *
    addr);

typedef PRInt16(*PRPollFN) (PRFileDesc * fd, PRInt16 in_flags,
    PRInt16 * out_flags);

struct PRIOMethods {
    PRDescType file_type;
    PRCloseFN close;
    PRReadFN read;
    PRAvailableFN available;
    PRAvailable64FN available64;
    PRFsyncFN fsync;
    PRSeekFN seek;
    PRSeek64FN seek64;
    PRFileInfoFN fileInfo;
    PRFileInfo64FN fileInfo64;
    PRAvailableFN write;
    PRSyncFN writev;
    PRConnectFN connect;
    PRAcceptFN accept;
    PRBindFN bind;
    PRListenFN listen;
    PRShutdownFN shutdown;
    PRRecvFN recv;
    PRSendFN send;
    PRRecvfromFN recvfrom;
    PRSendtoFN sendto;
    PRPollFN poll;
    PRAcceptreadFN acceptread;
    PRTransmitfileFN transmitfile;
    PRGetsocknameFN getsockname;
    PRGetpeernamFN getpeername;
    PRReservedFN reserved_fn_6;
    PRReservedFN reserved_fn_5;
    PRGetsocketoptionFN getsocketoption;
    PRSetsocketoptionFN setsocketoption;
    PRSendfileFN sendfile;
    PREFetchFileFN prefetch;
    PRConnectcontinueFN connectcontinue;
    PRReservedFN reserved_fn_3;
    PRReservedFN reserved_fn_2;
    PRReservedFN reserved_fn_1;
    PRReservedFN reserved_fn_0;
};

typedef PRStatus("PRGetpeernamFN") (PRFileDesc * fd, PRNetAddr *
typedef enum PRShutdownHow {
    PR_SHUTDOWN_RCV = 0,
    PR_SHUTDOWN_SEND = 1,
    PR_SHUTDOWN_BOTH = 2
} PRShutdownHow;

extern PRFileDesc *PR_Accept(PRFileDesc * fd, PRNetAddr * addr,
                               PRIntervalTime timeout);
extern PRStatus PR_Bind(PRFileDesc * fd, const PRNetAddr * addr);
extern PRStatus PR_Close(PRFileDesc * fd);
extern PRStatus PR_Connect(PRFileDesc * fd, const PRNetAddr * addr,
                           PRIntervalTime timeout);
extern PRStatus PR_ConnectContinue(PRFileDesc * fd, PRInt16 out_flags);
extern PRFileDesc *PR_CreateIOLayerStub(PRDescIdentity ident,
                                        const struct PRIOMethods *methods);
extern PRStatus PR_CreatePipe(PRFileDesc * *readPipe,
                               PRFileDesc * *writePipe);
extern const struct PRIOMethods *PR_GetDefaultIOMethods(void);
extern PRDescType PR_GetDescType(PRFileDesc * file);
extern PRDescIdentity PR_GetLayersIdentity(PRFileDesc * fd);
extern PRStatus PR_GetSocketOption(PRFileDesc * fd,
                                    PRSocketOptionData * data);
extern PRDescIdentity PR_GetUniqueIdentity(const char *layer_name);
extern PRStatus PR_Listen(PRFileDesc * fd, PRIntn backlog);
extern PRFileDesc *PR_Open(const char *name, PRIntn flags, PRIntn mode);
extern PRFileDesc *PR_OpenTCPSocket(PRIntn af);
extern PRFileDesc *PR_OpenUDPSocket(PRIntn af);
extern PRInt32 PR_Poll(PRPollDesc * pds, PRIntn npds,
                        PRIntervalTime timeout);
extern PRFileDesc *PR_PopIOLayer(PRFileDesc * fd_stack,
                                 PRDescIdentity id);
extern PRStatus PR_PushIOLayer(PRFileDesc * fd_stack,
                                PRDescIdentity id,
                                PRFileDesc * layer);
extern PRInt32 PR_Read(PRFileDesc * fd, void *buf, PRInt32 amount);
extern PRInt32 PR_Recv(PRFileDesc * fd, void *buf, PRInt32 amount,
                       PRIntn flags, PRIntervalTime timeout);
extern PRInt32 PR_RecvFrom(PRFileDesc * fd, void *buf, PRInt32 amount,
                           PRIntn flags, PRNetAddr * addr,
                           PRIntervalTime timeout);
extern PRInt32 PR_Send(PRFileDesc * fd, const void *buf, PRInt32 amount,
                      PRIntn flags, PRIntervalTime timeout);
extern PRInt32 PR_SendTo(PRFileDesc * fd, const void *buf, PRInt32 amount,
                        PRIntn flags, const PRNetAddr * addr,
                        PRIntervalTime timeout);
extern PRStatus PR_SetSocketOption(PRFileDesc * fd,
                                   PRSocketOptionData * data);
extern PRStatus PR_Shutdown(PRFileDesc * fd, PRShutdownHow how);
extern PRInt32 PR_Write(PRFileDesc * fd, void *buf, PRInt32 amount);

24.2.11 nspr4/private/pprio.h

#define pprio_h___
typedef PRInt32 PRosfd;
extern PRFileDesc *PR_ImportTCPSocket(PRosfd osfd);

### 24.2.12 nspr4/prlock.h

```c
#define prlock_h___
typedef struct PRLock PRLock;
extern void PR_DestroyLock(PRLock * lock);
extern void PR_Lock(PRLock * lock);
extern PRLock *PR_NewLock(void);
extern PRStatus PR_Unlock(PRLock * lock);
```

### 24.2.13 nspr4/prmem.h

```c
#define PR_NEW(_struct) ((_struct *) PR_MALLOC(sizeof(_struct)))
#define PR_NEWZAP(_struct)       ((_struct*)PR_Calloc(1, sizeof(_struct)))
#define PR_CALLOC(_size)        (PR_Calloc(1, (_size)))
#define PR_MALLOC(_bytes)       (PR_Malloc(_bytes))
#define PR_REALLOC(_ptr, _size) (PR_Realloc((_ptr), (_size)))
#define PR_FREEIF(_ptr) if (_ptr) PR_DELETE(_ptr)
#define PR_DELETE(_ptr) { PR_Free(_ptr); (_ptr) = NULL; }

extern void *PR_Calloc(PRUint32 nelem, PRUint32 elsize);
extern void PR_Free(void *ptr);
extern void *PR_Malloc(PRUint32 size);
extern void *PR_Realloc(void *ptr, PRUint32 size);
```

### 24.2.14 nspr4/prmon.h

```c
#define prmon_h___
typedef struct PRMonitor PRMonitor;
```

### 24.2.15 nspr4/prnetdb.h

```c
#define prnetdb_h___
#define PR_NetAddrFamily(addr)  ((addr)->raw.family)
#define PR_NetAddrInetPort(addr)        
        ((addr)->raw.family == PR_AF_INET6 ? (addr)->ipv6.port : (addr)->ipv4.port)

typedef struct PRHostEnt {
    char *h_name;
    char **h_aliases;
    PRInt32 h_addrtype;
    PRInt32 h_length;
    char **h_addr_list;
} PRHostEnt;

typedef struct PRAddrInfo PRAAddrInfo;
typedef enum PRNetAddrValue {
    PR_IpAddrNull,
    PR_IpAddrAny,
    PR_IpAddrLoopback,
    PR_IpAddrV4Mapped
} PRNetAddrValue;

typedef struct PRNetAddr {
    PRNetAddrValue va_family;
    char *service;
    PRInt32 va_persist;
    char *v4addr;
    char *v6addr;
} PRNetAddr;

extern void PR_ConvertIPv4AddrToIPv6(PRUint32 v4addr, PRIPv6Addr * v6addr);
```
extern void *PR_EnumerateAddrInfo(void *enumPtr,
    const PRAddrInfo * addrInfo,
    PRUint16 port, PRNetAddr * result);
extern void PR_FreeAddrInfo(PRAddrInfo * addrInfo);
extern PRAddrInfo *PR_GetAddrInfoByName(const char *hostname,
    PRUint16 af,
    PRIntn flags);
extern const char *PR_GetCanonNameFromAddrInfo(const PRAddrInfo * addrInfo);
extern PRStatus PR_InitializeNetAddr(PRNetAddrValue val, PRUint16 port,
    PRNetAddr * addr);
extern PRStatus PR_NetAddrToString(const PRNetAddr * addr, char *
    string,
    PRUint32 size);
extern PRStatus PR_StringToNetAddr(const char *string, PRNetAddr *
    addr);

24.2.16 nspr4/prthread.h

#define prthread_h___

typedef struct PRThread PRThread;
typedef enum PRThreadType {
    PR_USER_THREAD,
    PR_SYSTEM_THREAD
} PRThreadType;
typedef enum PRThreadScope {
    PR_LOCAL_THREAD,
    PR_GLOBAL_THREAD,
    PR_GLOBAL_BOUND_THREAD
} PRThreadScope;
typedef enum PRThreadState {
    PR_JOINABLE_THREAD,
    PR_UNJOINABLE_THREAD
} PRThreadState;
typedef enum PRThreadPriority {
    PR_PRIORITY_FIRST = 0,
    PR_PRIORITY_LOW = 0,
    PR_PRIORITY_NORMAL = 1,
    PR_PRIORITY_HIGH = 2,
    PR_PRIORITY_Urgent = 3,
    PR_PRIORITY_LAST = 3
} PRThreadPriority;
typedef void (*PRThreadPrivateDTOR) (void *priv);
extern PRStatus PR_ClearInterrupt(void);
extern void *PR_GetThreadPrivate(PRUintn tpdIndex);
extern PRThreadScope PR_GetThreadScope(const PRThread * thread);
extern PRThreadState PR_GetThreadState(const PRThread * thread);
extern PRStatus PR_Interrupt(PRThread * thread);
extern PRStatus PR_NewThreadPrivateIndex(PRUintn * newIndex,
    PRThreadPrivateDTOR destructor);
extern PRStatus PR_SetThreadPrivate(PRUintn tpIndex, void *priv);
extern PRStatus PR_Sleep(PRIntervalTime ticks);

24.2.17 nspr4/prtime.h

#define prtime_h___

typedef PRInt64 PRT ime;
typedef struct PRT imeParameters {
typedef struct PRExplodedTime {
    PRInt32 tm_usec;
    PRInt32 tm_sec;
    PRInt32 tm_min;
    PRInt32 tm_hour;
    PRInt32 tm_mday;
    PRInt32 tm_month;
    PRInt16 tm_year;
    PRInt8 tm_wday;
    PRInt16 tm_yday;
    PRTimeParameters tm_params;
} PRExplodedTime;

typedef PRTimeParameters(*PRTimeParamFn) (const PRExplodedTime * gmt);

extern void PR_ExplodeTime(PRTime usecs, PRTimeParamFn params,
                             PRExplodedTime * exploded);

extern PRUint32 PR_FormatTime(char *buf, int buflen, const char *fmt,
                               const PRExplodedTime * tm);

extern PRUint32 PR_GMTParameters(const PRExplodedTime * gmt);

extern PRTime PR_ImplodeTime(const PRExplodedTime * exploded);

extern PRTimeParameters PR_LocalTimeParameters(const PRExplodedTime * gmt);

extern void PR_NormalizeTime(PRExplodedTime * exploded,
                             PRTimeParamFn params);

extern PRTime PR_Now(void);

extern PRStatus PR_ParseTimeString(const char *string,
                                    PRBool default_to_gmt, PRTime
                                    * result);

extern PRStatus PR_ParseTimeStringToExplodedTime(const char *string,
                                                   PRBool default_to_gmt,
                                                   PRExplodedTime * result);

24.2.18 nspr4/prtypes.h

#define prtypes_h___

typedef int PRInt32;
typedef unsigned long int PRUword;
typedef int PRIntn;
typedef unsigned long int PRUint64;
typedef unsigned char PRUint8;
typedef short int PRInt16;
typedef long int PRInt64;
typedef PRIntn PRBool;
typedef unsigned short PRUint16;
typedef unsigned int PRUint32;
typedef size_t PRSize;
typedef unsigned int PRUintn;
typedef PRInt64 PROffset64;
typedef PRInt32 PROffset32;
typedef enum {
    PR_FAILURE = -1,
    PR_SUCCESS = 0
} PRStatus;
typedef signed char PRInt8;
24.3 Interfaces for libnss3

Table 24-3 defines the library name and shared object name for the libnss3 library

<table>
<thead>
<tr>
<th>Library:</th>
<th>libnss3</th>
</tr>
</thead>
<tbody>
<tr>
<td>SONAME:</td>
<td>libnss3.so</td>
</tr>
</tbody>
</table>

The behavior of the interfaces in this library is specified by the following specifications: [NSS SSL] Mozilla's NSS SSL Reference

24.3.1 NSS Utility

24.3.1.1 Interfaces for NSS Utility

An LSB conforming implementation shall provide the generic functions for NSS Utility specified in Table 24-4, with the full mandatory functionality as described in the referenced underlying specification.

Table 24-4 libnss3 - NSS Utility Function Interfaces

<table>
<thead>
<tr>
<th>CERT_CheckCertValidTimes(NSS_3.2) [NSS_SSL]</th>
<th>CERT_DestroyCertificate(NSS_3.2) [NSS_SSL]</th>
<th>CERT_DupCertificate(NSS_3.2) [NSS_SSL]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERT_FreeNicknames(NSS_3.2) [NSS_SSL]</td>
<td>CERT_GetCertNickname(NSS_3.2) [NSS_SSL]</td>
<td>CERT_GetDefaultCertificate(NSS_3.2) [NSS_SSL]</td>
</tr>
<tr>
<td>CERT_VerifyCertificate(NSS_3.2) [NSS_SSL]</td>
<td>CERT_VerifyCertificateNow(NSS_3.2) [NSS_SSL]</td>
<td>NSS_Init(NSS_3.2) [NSS_SSL]</td>
</tr>
<tr>
<td>NSS_InitReadWrite(NSS_3.2) [NSS_SSL]</td>
<td>NSS_NoDB_Init(NSS_3.2) [NSS_SSL]</td>
<td>NSS_Shutdown(NSS_3.2) [NSS_SSL]</td>
</tr>
<tr>
<td>PK11_FindCertFromNickname(NSS_3.2) [NSS_SSL]</td>
<td>PK11_FindKeyByAnyCertificate(NSS_3.2) [NSS_SSL]</td>
<td>PK11_GetSlotName(NSS_3.2) [NSS_SSL]</td>
</tr>
<tr>
<td>PK11_GetTokenName(NSS_3.2) [NSS_SSL]</td>
<td>PK11_IsHW(NSS_3.2) [NSS_SSL]</td>
<td>PK11_IsPresent(NSS_3.2) [NSS_SSL]</td>
</tr>
<tr>
<td>PK11_IsReadOnly(NSS_3.2) [NSS_SSL]</td>
<td>PK11_SetPasswordFunction(NSS_3.2) [NSS_SSL]</td>
<td>SECKEY_DestroyPrivateKey(NSS_3.2) [NSS_SSL]</td>
</tr>
</tbody>
</table>

24.4 Data Definitions for libnss3

This section defines global identifiers and their values that are associated with interfaces contained in libnss3. 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.
24.4.1 nss3/blapit.h

#define _BLAPIT_H_
#define PQG_PBITS_TO_INDEX(bits)         
  (((bits) < 512 || (bits) > 1024 || (bits) % 64) ? -1 : (int)((bits)-512)/64)
#define PQG_INDEX_TO_PBITS(j)    
  (((unsigned)(j) > 8) ? -1 : (512 + 64 * (j)))
#define NSS_AES 0
#define NSS_DES 0
#define NSS_RC2 0
#define NSS_RC5 0
#define NSS_AES_CBC 1
#define NSS_DES_CBC 1
#define NSS_RC2_CBC 1
#define NSS_RC5_CBC 1
#define DSA_MAX_P_BITS 1024
#define DH_MIN_P_BITS 128
#define RSA_MIN_MODULUS_BITS 128
#define AES_BLOCK_SIZE 16
#define DSA_Q_BITS 160
#define NSS_DES_EDE3 2
#define DSA_SUBPRIME_LEN 20
#define NSS_FREEBL_DEFAULT_CHUNKSIZE 2048
#define DH_MAX_P_BITS 2236
#define NSS_DES_EDE3_CBC 3
#define DSA_SIGNATURE_LEN 40
#define DSA_MIN_P_BITS 512
#define AES_KEY_WRAP_BLOCK_SIZE 8
#define AES_KEY_WRAP_IV_BYTES 8
#define DES_KEY_LENGTH 8

typedef struct PQGParamsStr {
    PLArenaPool *arena;
    SECItem prime;
    SECItem subPrime;
    SECItem base;
} PQGParams;

typedef struct PQGVerifyStr {
    PLArenaPool *arena;
    unsigned int counter;
    SECItem seed;
    SECItem h;
} PQGVerify;

24.4.2 nss3/cert.h

#define _CERT_H_
extern SECCertTimeValidity CERT_CheckCertValidTimes(CERTCertificate * cert, PRTime t, PRBool allowOverride);
extern void CERT_DestroyCertificate(CERTCertificate * cert);
extern CERTCertificate *CERT_DupCertificate(CERTCertificate * c);
extern void CERT_FreeNicknames(CERTCertNicknames * nicknames);
extern CERTCertNicknames *CERT_GetCertNicknames(CERTCertDBHandle * handle, int what, void *wincx);
extern CERTCertDBHandle *CERT_GetDefaultCertDB(void);
extern SECStatus CERT_VerifyCertName(CERTCertificate * cert,
const char *hostname);

extern SECStatus CERT_VerifyCertNow(CERTCertDBHandle * handle,
CERTCertificate * cert,
PRBool checkSig,
SECCertUsage certUsage, void
*wincx);

24.4.3 nss3/certt.h

#define _CERTT_H_
#define NS_CERT_TYPE_CA  
   ( NS_CERT_TYPE_SSL_CA | NS_CERT_TYPE_EMAIL_CA |  
NS_CERT_TYPE_OBJECT_SIGNING_CA  |  
EXT_KEY_USAGE_STATUS_RESPONDER )
#define NS_CERT_TYPE_APP  
   ( NS_CERT_TYPE_SSL_CLIENT | NS_CERT_TYPE_SSL_SERVER |  
NS_CERT_TYPE_EMAIL | NS_CERT_TYPE_OBJECT_SIGNING )
#define SEC_GET_TRUST_FLAGS(trust,type)  
   (((type)==trustSSL)?((trust)->sslFlags):  
   (((type)==trustEmail)?((trust)->emailFlags):  
   (((type)==trustObjectSigning)?((trust)->  
   objectSigningFlags):0)))
#define KU_ALL   
   (KU_DIGITAL_SIGNATURE | KU_NON_REPUDIATION |  
KU_KEY_ENCIIPHERMENT | KU_DATA_ENCIIPHERMENT | KU_KEY_AGREEMENT |  
KU_CRL_SIGN)  
#define CERT_LIST_END(n,l)      (((void *)n) == ((void *)&l-  
>list))
#define CERT_LIST_NEXT(n)        ((CERTCertListNode  *)n-  
>links.next)
#define CERT_LIST_HEAD(l)        ((CERTCertListNode  
)>PR_LIST_HEAD(&l->list))
#define certificateUsageSSLClient       (0x0001)  
#define certificateUsageSSLServer       (0x0002)  
#define certificateUsageSSLServerWithStepUp     (0x0004)  
#define certificateUsageSSLCA   (0x0008)  
#define certificateUsageEmailSigner     (0x0010)  
#define certificateUsageEmailRecipient  (0x0020)  
#define certificateUsageObjectSigner    (0x0040)  
#define certificateUsageUserCertImport  (0x0080)  
#define certificateUsageVerifyCA        (0x0100)  
#define KU_CRL_SIGN   (0x02)  
#define NS_CERT_TYPE_OBJECT_SIGNING_CA  (0x04)  
#define certificateUsageProtectedObjectSigner (0x0200)
#define KU_KEY_CERT_SIGN (0x04)  
#define NS_CERT_TYPE_SSL_CA     (0x08)  
#define KU_CESSATION_OF_OPERATION       (0x04)  
#define certificateUsageStatusResponder (0x0400)
#define KU_KEY_AGREEMENT (0x08)  
#define NS_CERT_TYPE_RESERVED (0x08)  
#define RF_CERTIFICATE_COMPROMISE (0x08)
#define certificateUsageAnyCA (0x08)
#define KU_DATA_ENCIIPHERMENT (0x10)  
#define NS_CERT_TYPE_OBJECT_SIGNING (0x10)  
#define RF_AFFILIATION_CHANGED (0x10)  
#define KU_KEY_ENCIIPHERMENT (0x20)  
#define NS_CERT_TYPE_EMAIL (0x20)  
#define RF_CA_COMPROMISE (0x20)  
#define KU_NON_REPUDIATION (0x40)  
#define NS_CERT_TYPE_SSL_SERVER (0x40)  
#define RF_KEY_COMPROMISE (0x40)
#define EXT_KEY_USAGE_STATUS_RESPONDER (0x4000)
#define KU_KEY_AGREEMENT_OR_ENCIPHERMENT (0x4000)
#define KU_DIGITAL_SIGNATURE (0x80)
#define NS_CERT_TYPE_SSL_CLIENT (0x80)
#define RF_UNUSED (0x80)
#define EXT_KEY_USAGE_TIME_STAMP (0x8000)
#define KU_NS_GOVT_APPROVED (0x8000)
#define CERT_UNLIMITED_PATH_CONSTRAINT -2
#define SEC_CERTIFICATE_REQUEST_VERSION 0
#define SEC_CERTIFICATE_VERSION_1 0
#define SEC_CRL_VERSION_1 0
#define SEC_CERTIFICATE_VERSION_2 1
#define SEC_CERT_CLASS_CA 1
#define SEC_CERT_NICKNAMES_ALL 1
#define SEC_CRL_VERSION_2 1
#define SEC_CERTIFICATE_VERSION_3 2
#define SEC_CERT_CLASS_SERVER 2
#define SEC_CERT_NICKNAMES_USER 2
#define SEC_MAX_CERT_CHAIN 20
#define SEC_CERT_CLASS_USER 3
#define SEC_CERT_NICKNAMES_SERVER 3
#define SEC_CERT_CLASS_EMAIL 4
#define SEC_CERT_NICKNAMES_CA 4
#define certificateUsageHighest certificateUsageAnyCA
#define CERT_LIST_EMPTY(l) CERT_LIST_END(CERT_LIST_HEAD(l), l)

typedef struct CERTAVAStr {
  SECItem type;
  SECItem value;
} CERTAVA;

typedef struct CERTAttributeStr {
  SECItem attrType;
  SECItem **attrValue;
} CERTAttribute;

typedef struct CERTAuthInfoAccessStr {
  SECItem method;
  SECItem derLocation;
  CERTGeneralName *location;
} CERTAuthInfoAccess;

typedef struct CERTAuthKeyIDStr {
  SECItem keyID;
  CERTGeneralName *authCertIssuer;
  SECItem authCertSerialNumber;
  SECItem **DERAuthCertIssuer;
} CERTAuthKeyID;

typedef struct CERTBasicConstraintsStr {
  PRBool isCA;
  int pathLenConstraint;
} CERTBasicConstraints;

typedef struct NSSSTrustDomainStr CERTCertDBHandle;

typedef struct CERTCertExtensionStr {
  SECItem id;
  SECItem critical;
  SECItem value;
} CERTCertExtension;

typedef struct CERTCertListNodeStr {
  PRCList links;
  CERTCertificate *cert;
  void *appData;
} CERTCertListNode;

typedef struct CERTCertNicknamesStr {

typedef struct CERTCertTrustStr {
    unsigned int sslFlags;
    unsigned int emailFlags;
    unsigned int objectSigningFlags;
} CERTCertTrust;

typedef struct CERTSignedDataStr {
    SECItem data;
    SECAlgorithmID signatureAlgorithm;
    SECItem signature;
} CERTSignedData;

typedef struct CERTCertificateListStr {
    SECItem *certs;
    int len;
    PLArenaPool *arena;
} CERTCertificateList;

typedef struct CERTNameStr {
    PLArenaPool *arena;
    CERTRDN **rdns;
} CERTName;

typedef struct CERTCrlStr {
    PLArenaPool *arena;
    SECItem version;
    SECAlgorithmID signatureAlg;
    SECItem derName;
    CERTName name;
    SECItem lastUpdate;
    SECItem nextUpdate;
    CERTCrlEntry **entries;
    CERTCertExtension **extensions;
} CERTCrl;

typedef struct CERTCrlDistributionPointsStr {
    CRLDistributionPoint **distPoints;
} CERTCrlDistributionPoints;

typedef struct CERTCrlEntryStr {
    SECItem serialNumber;
    SECItem revocationDate;
    CERTCertExtension **extensions;
} CERTCrlEntry;

typedef struct CERTCrlHeadNodeStr {
    PLArenaPool *arena;
    CERTCrlDBHandle *dbHandle;
    CERTCrlNode *first;
    CERTCrlNode *last;
} CERTCrlHeadNode;

typedef struct CERTCrlNodeStr {
    CERTCrlNode *next;
    int type;
    CERTSignedCrl *crl;
} CERTCrlNode;

typedef struct CERTDistNamesStr {
    PLArenaPool *arena;
    int nnames;
    SECItem *names;
    void *head;
} CERTDistNames;

typedef struct OtherNameStr {
    SECItem name;
    SECItem oid;
} OtherName;
typedef struct CERTGeneralNameListStr {
  PLArenaPool *arena;
  CERTGeneralName *name;
  int refCount;
  int len;
  PRLock *lock;
} CERTGeneralNameList;

typedef struct CERTIssuerAndSNStr {
  SECItem derIssuer;
  CERTName issuer;
  SECItem serialNumber;
} CERTIssuerAndSN;

typedef struct CERTSubjectPublicKeyInfoStr {
  PLArenaPool *arena;
  SECAlgorithmID algorithm;
  SECItem subjectPublicKey;
} CERTSubjectPublicKeyInfo;

typedef struct CERTGeneralNameStr {
  CERTGeneralNameType type;
  union {
    CERTName directoryName;
    OtherName OthName;
    SECItem other;
  } name;
  SECItem derDirectoryName;
  PRLList l;
} CERTGeneralName;

typedef struct CERTNameConstraintsStr {
  CERTNameConstraint *permitted;
  CERTNameConstraint *excluded;
  SECItem **DERPermitted;
  SECItem **DERExcluded;
} CERTNameConstraints;

typedef struct CERTOKDomainNameStr {
  CERTOKDomainName *next;
  char name[1];
} CERTOKDomainName;

typedef struct CERTPrivKeyUsagePeriodStr {
  SECItem notBefore;
  SECItem notAfter;
  PLArenaPool *arena;
} CERTPrivKeyUsagePeriod;

typedef struct CERTRDNStr {
  CERTAVA **avas;
} CERTRDN;

typedef struct CERTSignedCrlStr {
  PLArenaPool *arena;
  CERTCrl crl;
  void *reserved1;
  PRBool reserved2;
  PRBool isperm;
  PRBool istemp;
  int referenceCount;
  CERTCertDBHandle *dbhandle;
  CERTSignedData signatureWrap;
  char *url;
  SECItem *derCrl;
  PK11SlotInfo *slot;
  CK_OBJECT_HANDLE pkcs11ID;
  void *opaque;
} CERTSignedCrl;

typedef struct CERTValidityStr {
  PLArenaPool *arena;
  SECItem notBefore;
  SECItem notAfter;
} CERTValidity;
typedef struct CERTStatusConfigStr {
    CERTStatusChecker statusChecker;
    CERTStatusDestroy statusDestroy;
    void *statusContext;
} CERTStatusConfig;

typedef struct CERTSubjectListStr {
    PLArenaPool *arena;
    int ncerts;
    char *emailAddr;
    CERTSubjectNode *head;
    CERTSubjectNode *tail;
    void *entry;
} CERTSubjectList;

typedef struct CERTSubjectNodeStr {
    struct CERTSubjectNodeStr *next;
    struct CERTSubjectNodeStr *prev;
    SECItem certKey;
    SECItem keyID;
} CERTSubjectNode;

typedef struct CERTCertificateRequestStr {
    PLArenaPool *arena;
    SECItem version;
    CERTName subject;
    CERTSubjectPublicKeyInfo subjectPublicKeyInfo;
    CERTAttribute **attributes;
} CERTCertificateRequest;

typedef struct CERTCertificateStr {
    PLArenaPool *arena;
    char *subjectName;
    char *issuerName;
    CERTSignedData signatureWrap;
    SECItem derCert;
    SECItem derIssuer;
    SECItem derSubject;
    SECItem derPublicKey;
    SECItem certKey;
    SECItem version;
    SECItem serialNumber;
    SECAlgorithmID signature;
    CERTName issuer;
    CERTValidity validity;
    CERTName subject;
    CERTSubjectPublicKeyInfo subjectPublicKeyInfo;
    SECItem issuerID;
    SECItem subjectID;
    CERTCertExtension **extensions;
    char *emailAddr;
    CERTCertDBHandle *dbhandle;
    SECItem subjectKeyID;
    PRBool keyIDGenerated;
    unsigned int keyUsage;
    unsigned int rawKeyUsage;
    PRBool keyUsagePresent;
    PRUint32 nsCertType;
    PRBool keepSession;
    PRBool timeOK;
    CERTOKDomainName *domainOK;
    PRBool isperm;
    PRBool istemp;
    char *nickname;
    char *dbnickname;
    struct NSSCertificateStr *nssCertificate;
    CERTCertTrust *trust;
    int referenceCount;
    CERTSubjectList *subjectList;
    CERTAuthKeyID *authKeyID;
PRBool isRoot;
union {
    void *apointer;
    struct {
        unsigned int hasUnsupportedCriticalExt;
    } bits;
} options;
int series;
PK11SlotInfo *slot;
CK_OBJECT_HANDLE pkcs11ID;
PRBool ownSlot;
}
CERTCertificate;
typedef struct CERTVerifyLogStr {
    PLArenaPool *arena;
    unsigned int count;
    struct CERTVerifyLogNodeStr *head;
    struct CERTVerifyLogNodeStr *tail;
} CERTVerifyLog;
typedef struct CRLDistributionPointStr {
    DistributionPointTypes distPointType;
    union {
        CERTGeneralName *fullName;
        CERTRDN relativeName;
    } distPoint;
    SECItem reasons;
    CERTGeneralName *crlIssuer;
    SECItem derDistPoint;
    SECItem derRelativeName;
    SECItem *derCrlIssuer;
    SECItem **derFullName;
    SECItem bitmap;
} CRLDistributionPoint;
typedef enum SECCertUsageEnum {
    certUsageSSLClient,
    certUsageSSLServer = 1,
    certUsageSSLServerWithStepUp = 2,
    certUsageSSLCA = 3,
    certUsageEmailSigner = 4,
    certUsageEmailRecipient = 5,
    certUsageObjectSigner = 6,
    certUsageUserCertImport = 7,
    certUsageVerifyCA = 8,
    certUsageProtectedObjectSigner = 9,
    certUsageStatusResponder = 10,
    certUsageAnyCA = 11
} SECCertUsage;
typedef PRInt64 SECCertificateUsage;
typedef enum SECCertTimeValidityEnum {
    secCertTimeValid,
    secCertTimeExpired = 1,
    secCertTimeNotValidYet = 2,
    secCertTimeUndetermined = 3
} SECCertTimeValidity;
typedef enum CERTCompareValidityStatusEnum {
    certValidityUndetermined,
    certValidityChooseB = 1,
    certValidityEqual = 2,
    certValidityChooseA = 3
} CERTCompareValidityStatus;
typedef enum CERTGeneralNameTypeEnum {
    certOtherName = 1,
    certRFC822Name = 2,
    certDNSName = 3,
    certX400Address = 4,
    certDirectoryName = 5,
    certEDIPartyName = 6,
typedef struct CERTNameConstraintStr {
    CERTGeneralName name;
    SECItem DERName;
    SECItem min;
    SECItem max;
    PRCList l;
} CERTNameConstraint;

typedef enum DistributionPointTypesEnum {
    generalName = 1,
    relativeDistinguishedName = 2
} DistributionPointTypes;

struct CERTVerifyLogNodeStr {
    CERTCertificate *cert;
    long int error;
    unsigned int depth;
    void *arg;
    struct CERTVerifyLogNodeStr *next;
    struct CERTVerifyLogNodeStr *prev;
};

typedef SECStatus(*CERTStatusChecker) (CERTCertDBHandle *,
    CERTCertificate *,
    PRInt64, void *);

typedef SECStatus(*CERTStatusDestroy) (CERTStatusConfig *);

typedef struct {
    SECOidTag oid;
    SECItem qualifierID;
    SECItem qualifierValue;
} CERTPolicyQualifier;

typedef struct {
    SECOidTag oid;
    SECItem policyID;
    CERTPolicyQualifier **policyQualifiers;
} CERTPolicyInfo;

typedef struct {
    PLArenaPool *arena;
    CERTPolicyInfo **policyInfos;
} CERTCertificatePolicies;

typedef struct {
    SECItem organization;
    SECItem **noticeNumbers;
} CERTNoticeReference;

typedef struct {
    PLArenaPool *arena;
    CERTNoticeReference noticeReference;
    SECItem derNoticeReference;
    SECItem displayText;
} CERTUserNotice;

typedef struct {
    PLArenaPool *arena;
    SECItem **oids;
} CERTOidSequence;

#### 24.4.4 nss3/cmsreclist.h

#define _CMSRECLIST_H

typedef struct NSSCMSRecipientStr {
    int riIndex;
    int subIndex;
    enum {
        RLIssuerSN,
RLSubjKeyID = 1
} kind;
union {
    CERTIssuerAndSN *issuerAndSN;
    SECIItem *subjectKeyID;
} id;
CERTCertificate *cert;
SECKeyPrivateKey *privkey;
PK11SlotInfo *slot;
} NSSCMSRecipient;

### 24.4.5 nss3/cryptoht.h

#define _CRYPTOHT_H_

typedef struct SGNContextStr SGNContext;
typedef struct VFYContextStr VFYContext;

### 24.4.6 nss3/hasht.h

#define _HASHT_H_
#define MD2_LENGTH 16
#define MD5_LENGTH 16
#define SHA1_LENGTH 20
#define SHA256_LENGTH 32
#define SHA384_LENGTH 48
#define SHA512_LENGTH 64
#define HASH_LENGTH_MAX SHA512_LENGTH

typedef struct SECHashObjectStr {
    unsigned int length;
    void *("create") (void);
    void *("clone") (void *");
    void *("destroy") (void *, PRBool);
    void *("begin") (void *");
    void *("update") (void *, const unsigned char *, unsigned int);
    void *("end") (void *, unsigned char *, unsigned int *,
    unsigned int);
    unsigned int blocklength;
    HASH_HashType type;
} SECHashObject;
typedef struct HASHContextStr {
    const struct SECHashObjectStr *hashobj;
    void *hash_context;
} HASHContext;
typedef enum {
    HASH_AlgNULL,
    HASH_AlgMD2 = 1,
    HASH_AlgMD5 = 2,
    HASH_AlgSHA1 = 3,
    HASH_AlgSHA256 = 4,
    HASH_AlgSHA384 = 5,
    HASH_AlgSHA512 = 6,
    HASH_AlgTOTAL = 7
} HASH_HashType;

### 24.4.7 nss3/key.h

#define _KEY_H_

© 2015 Linux Foundation
24.4.8 nss3/keyhi.h

#define _KEYHI_H_
extern void SECKEY_DestroyPrivateKey(SECKEYPrivateKey * key);

24.4.9 nss3/keyt.h

#define _KEYT_H_

24.4.10 nss3/keythi.h

#define _KEYTHI_H_

typedef enum {
    nullKey,
    rsaKey = 1,
    dsaKey = 2,
    forteezaKey = 3,
    dhKey = 4,
    keaKey = 5,
    ecKey = 6
} KeyType;
typedef struct SECKEYRSAPublicKeyStr {
    PLArenaPool *arena;
    SECItem modulus;
    SECItem publicExponent;
} SECKEYRSAPublicKey;
typedef struct SECKEYPQGParamsStr {
    PLArenaPool *arena;
    SECItem prime;
    SECItem subPrime;
    SECItem base;
} SECKEYPQGParams;
typedef struct SECKEYDSAPublicKeyStr {
    SECKEYPQGParams params;
    SECItem publicValue;
} SECKEYDSAPublicKey;
typedef struct SECKEYDHParamsStr {
    PLArenaPool *arena;
    SECItem prime;
    SECItem base;
} SECKEYDHParams;
typedef struct SECKEYDHPublicKeyStr {
    PLArenaPool *arena;
    SECItem prime;
    SECItem base;
    SECItem publicValue;
} SECKEYDHPublicKey;
typedef SECItem SECKEYECPParams;
typedef struct SECKEYECPublicKeyStr {
    SECKEYECPParams DEREncodedParams;
    int size;
    SECItem publicValue;
} SECKEYECPublicKey;
typedef struct SECKEYFortezzaPublicKeyStr {
    int KEAversion;
    int DSSversion;
    unsigned char KMID[8];
    SECItem clearance;
    SECItem KEApriviledge;
    SECItem DSSpriviledge;
typedef struct SECKEYKEAParamsStr {
    PLArenaPool *arena;
    SECItem hash;
} SECKEYKEAParams;

typedef struct SECKEYKEAPublicKeyStr {
    SECKEYKEAParams params;
    SECItem publicValue;
} SECKEYKEAPublicKey;

typedef struct SECKEYPublicKeyStr {
    PLArenaPool *arena;
    KeyType keyType;
    PK11SlotInfo *pkcs11Slot;
    CK_OBJECT_HANDLE pkcs11ID;
    union {
        SECKEYRSAPublicKey rsa;
        SECKEYDSAPublicKey dsa;
        SECKEYDH PublicKey dh;
        SECKEYKEAPublicKey kea;
        SECKEYFortezzaPublicKey fortezza;
        SECKEYECPublicKey ec;
    } u;
} SECKEYPublicKey;

typedef struct SECKEYPrivateKeyStr {
    PLArenaPool *arena;
    KeyType keyType;
    PK11SlotInfo *pkcs11Slot;
    CK_OBJECT_HANDLE pkcs11ID;
    PRBool pkcs11IsTemp;
    void *wincx;
    PRUint32 staticflags;
} SECKEYPrivateKey;

typedef struct {
    PRCList list;
    SECKEYPrivateKey *key;
} SECKEYPrivateKeyListNode;

typedef struct {
    PRCList list;
    PLArenaPool *arena;
} SECKEYPrivateKeyList;

typedef struct {
    PRCList list;
    SECKEYPrivateKeyList *key;
} SECKEYPrivateKeyListNode;

typedef struct {
    PRCList list;
    PLArenaPool *arena;
} SECKEYPublicKeyList;

typedef struct {
    PRCList list;
} SECKEYPublicKeyListNode;

24.4.11 nss3/nss.h

#define __nss_h_
#define NSS_VERSION     "3.11.4"
#define NSS_INIT_READONLY       0x1
#define NSS_INIT_NOROOTINIT     0x10
#define NSS_INIT_NOPK11FINALIZE 0x100
#define NSS_INIT_NOCERTDB       0x2
#define NSS_INIT_OPTIMIZESPACE  0x20
#define NSS_INIT_RESERVED       0x200
#define NSS_INIT_NOMODDB        0x4
#define NSS_INIT_PK11THREADSAFE 0x40
#define NSS_INIT_FORCEOPEN      0x8
#define NSS_INIT_PK11RELOAD     0x80
#define NSS_VMINOR      11
#define NSS_VMAJOR      3
#define NSS_VPATCH 4
#define NSS_INIT_COOPERATE | NSS_INIT_PK11THREADSAFE | NSS_INIT_PK11RELOAD | NSS_INIT_NOPK11FINALIZE | NSS_INIT_RESERVED
#define SECMOD_DB "secmod.db"

extern SECStatus NSS_Init(const char *configdir);
extern SECStatus NSS_InitReadWrite(const char *configdir);
extern SECStatus NSS_NoDB_Init(const char *configdir);
extern SECStatus NSS_Shutdown(void);

24.4.12 nss3/nssb64.h

#define _NSSB64_H_

24.4.13 nss3/nssb64t.h

#define _NSSB64T_H_

typedef struct NSSBase64DecoderStr NSSBase64Decoder;
typedef struct NSSBase64EncoderStr NSSBase64Encoder;

24.4.14 nss3/nssilckt.h

#define _NSSILCKT_H_

typedef enum {
    nssILockArena,
    nssILockSession = 1,
    nssILockObject = 2,
    nssILockRefLock = 3,
    nssILockCert = 4,
    nssILockCertDB = 5,
    nssILockDBM = 6,
    nssILockCache = 7,
    nssILockSSL = 8,
    nssILockList = 9,
    nssILockSlot = 10,
    nssILockFreelist = 11,
    nssILockOID = 12,
    nssILockAttribute = 13,
    nssILockPK11cxt = 14,
    nssILockRWLock = 15,
    nssILockOther = 16,
    nssILockSelfServ = 17,
    nssILockKeyDB = 18,
    nssILockLast = 19
} nssILockType;

24.4.15 nss3/nssrwlkt.h

#define nssrwlkt_h___

typedef struct nssRWLockStr NSSRWLock;

24.4.16 nss3/ocspt.h

#define _OCSPT_H_
typedef struct CERTOCSPRequestStr CERTOCSPRequest;
typedef struct CERTOCSPResponseStr CERTOCSPResponse;
typedef struct CERTOCSPCertIDStr CERTOCSPCertID;
typedef struct CERTOCSPSingleResponseStr CERTOCSPSingleResponse;

24.4.17 nss3/pk11pub.h

#define _PK11PUB_H_

extern CERTCertificate *PK11_FindCertFromNickname(const char *nickname, 
                                                    void *wincx);
extern SECKEYPrivateKey *PK11_FindKeyByAnyCert(CERTCertificate * 
                                                    cert, 
                                                    void *wincx);
extern char *PK11_GetSlotName(PK11SlotInfo * slot);
extern char *PK11_GetTokenName(PK11SlotInfo * slot);
extern PRBool PK11_IsHW(PK11SlotInfo * slot);
extern PRBool PK11_IsPresent(PK11SlotInfo * slot);
extern PRBool PK11_IsReadOnly(PK11SlotInfo * slot);
extern void PK11_SetPasswordFunc(PK11PasswordFunc func);

24.4.18 nss3/pkcs11t.h

#define _PKCS11T_H_

typedef unsigned char CK_BYTE;
typedef CK_BYTE CK_CHAR;
typedef CK_BYTE CK_UTF8CHAR;
typedef unsigned long int CK_ULONG;
typedef CK_ULONG CK_FLAGS;
typedef void *CK_VOID_PTR;
typedef struct CK_VERSION {
    CK_BYTE major;
    CK_BYTE minor;
} CK_VERSION;

typedef struct CK_INFO {
    CK_VERSION cryptokiVersion;
    CK_UTF8CHAR manufacturerID[31];
    CK_FLAGS flags;
    CK_UTF8CHAR libraryDescription[31];
    CK_VERSION libraryVersion;
} CK_INFO;

typedef CK_ULONG CK_SLOT_ID;

typedef struct CK_SLOT_INFO {
    CK_UTF8CHAR slotDescription[63];
    CK_UTF8CHAR manufacturerID[31];
    CK_FLAGS flags;
    CK_VERSION hardwareVersion;
    CK_VERSION firmwareVersion;
} CK_SLOT_INFO;

typedef struct CK_TOKEN_INFO {
    CK_UTF8CHAR label[31];
    CK_UTF8CHAR manufacturerID[31];
    CK_UTF8CHAR model[15];
    CK_CHAR serialNumber[15];
    CK_FLAGS flags;
    CK_ULONG ulMaxSessionCount;
    CK_ULONG ulSessionCount;
    CK_ULONG ulMaxRwSessionCount;
    CK_ULONG ulRwSessionCount;
    CK_ULONG ulMaxPinLen;
    CK_ULONG ulMinPinLen;
```c
CK_ULONG ulTotalPublicMemory;
CK_ULONG ulFreePublicMemory;
CK_ULONG ulTotalPrivateMemory;
CK_ULONG ulFreePrivateMemory;
CK_VERSION hardwareVersion;
CK_VERSION firmwareVersion;
CK_CHAR utcTime[15];
}
```
24.4.21 nss3/seccomon.h

#define _SECCOMMON_H_

typedef enum {
    siBuffer,
    siClearDataBuffer = 1,
    siCipherDataBuffer = 2,
    siDERCertBuffer = 3,
    siEncodedCertBuffer = 4,
    siDERNameBuffer = 5,
    siEncodedNameBuffer = 6,
    siAsciiNameString = 7,
    siAsciiString = 8,
    siDEROID = 9,
    siUnsignedInteger = 10,
    siUTCTime = 11,
    siGeneralizedTime = 12,
    siVisibleString = 13,
    siUTF8String = 14,
    siBMPString = 15
} SECItemType;

typedef struct SECItemStr {
    SECItemType type;
    unsigned char *data;
    unsigned int len;
} SECItem;

typedef enum _SECStatus {
    SECWouldBlock = -2,
    SECFailure = -1,
    SECSuccess
} SECStatus;

typedef enum _SECComparison {
    SECLessThan = -1,
    SECEqual,
    SECGreaterThan = 1
} SECComparison;

24.4.22 nss3/secdert.h

#define _SECDERT_H_

typedef struct DERTemplateStr {
    unsigned long int kind;
    unsigned int offset;
    DERTemplate *sub;
    unsigned long int arg;
} DERTemplate;

24.4.23 nss3/secdigt.h

#define _SECDIGT_H_

typedef struct SGNDigestInfoStr {
    PLArenaPool *arena;
    SECAlgorithmID digestAlgorithm;
    SECItem digest;
} SGNDigestInfo;
#define _SECMOD_T_H_
#define SECMOD_MAKE_NSS_FLAGS(fips,slot)         
"Flags=internal,critical"fips" 
slotparams="{"slot"="SECMOD_SLOT_FLAGS"})"
#define SECMOD_FIPS_NAME "NSS Internal FIPS PKCS #11 Module"
#define SECMOD_INT_NAME "NSS Internal PKCS #11 Module"
#define SECMOD_SLOT_FLAGS "slotFlags=[RSA,DSA,DH,RC2,RC4,RANDOM,SHA1,MD5,MD2,SSL,TLS,AES,SHA256,SHA512]"
#define SECMODEXTERNAL 0
#define CRL_IMPORT_DEFAULT_OPTIONS 0x00000000
#define CRL_IMPORT_BYPASS_CHECKS 0x00000001
#define PK11_ATTR_TOKEN 0x00000001L
#define PK11_ATTR_SESSION 0x00000002L
#define PK11_ATTR_PRIVATE 0x00000004L
#define PK11_ATTR_PUBLIC 0x00000008L
#define PK11_ATTR_MODIFIABLE 0x00000010L
#define PK11_ATTR_UNMODIFIABLE 0x00000020L
#define PK11_ATTR_Sensitive 0x00000040L
#define PK11_ATTR_INSENSITIVE 0x00000080L
#define PK11_ATTR_EXTRACTABLE 0x00000100L
#define PK11_ATTR_UNEXTRACTABLE 0x00000200L
#define PK11_ATTR_SENSITIVE 0x00000400L
#define PK11_ATTR_INSENSITIVE 0x00000800L
#define PK11_ATTR_EXTRACTABLE 0x00001000L
#define PK11_ATTR_UNEXTRACTABLE 0x00002000L
#define PK11_ATTR_SENSITIVE 0x00004000L
#define PK11_ATTR_INSENSITIVE 0x00008000L
#define PK11_ATTR_MODIFIABLE 0x00010000L
#define PK11_ATTR_UNMODIFIABLE 0x00020000L
#define PK11_ATTR_Sensitive 0x00040000L
#define PK11_ATTR_INSENSITIVE 0x00080000L
#define PK11_ATTR_EXTRACTABLE 0x00100000L
#define PK11_ATTR_UNEXTRACTABLE 0x00200000L
#define PK11_ATTR_SENSITIVE 0x00400000L
#define PK11_ATTR_INSENSITIVE 0x00800000L
#define PK11_ATTR_MODIFIABLE 0x01000000L
#define PK11_ATTR_UNMODIFIABLE 0x02000000L
#define PK11_ATTR_Sensitive 0x04000000L
#define PK11_ATTR_INSENSITIVE 0x08000000L
#define PK11_ATTR_EXTRACTABLE 0x10000000L
#define PK11_ATTR_UNEXTRACTABLE 0x20000000L
#define PK11_ATTR_SENSITIVE 0x40000000L
#define PK11_ATTR_INSENSITIVE 0x80000000L
#define CKM_FAKE_RANDOM 0x80000efeL
#define CKM_INVALID_MECHANISM 0xffffffffL
#define SECMOD_INTERNAL 1
#define SECMOD FIPS 2
#define PK11_PW_AUTHENTICATED "AUTH"
#define PK11_PW_RETRY "RETRY"
#define SECMOD_INT_FLAGS SECMOD_MAKE_NSS_FLAGS("",1)
#define SECMOD_FIPS_FLAGS SECMOD_MAKE_NSS_FLAGS("",fips",3)
#define PK11_PW_TRY "TRY"

typedef struct SECMODModuleStr {
    PLArenaPool *arena;
    PRBool internal;
    PRBool loaded;
    PRBool isFIPS;
    char *dllName;
}
char *commonName;
void *library;
void *functionList;
PRLock *refLock;
int refCount;
PK11SlotInfo **slots;
int slotCount;
PK11PreSlotInfo *slotInfo;
int slotInfoCount;
SECMODModuleID moduleID;
PRBool isThreadSafe;
unsigned long int ssl[1];
char *libraryParams;
void *moduleDBFunc;
SECMODModule *parent;
PRBool isCritical;
PRBool isModuleDB;
PRBool moduleDBOnly;
int trustOrder;
int cipherOrder;
unsigned long int evControlMask;
CK_VERSION cryptokiVersion;
} SECMODModule;
typedef struct SECMODModuleListStr {
    SECMODModuleList *next;
    SECMODModule *module;
} SECMODModuleList;
typedef NSSRWLock SECMODListLock;
typedef struct PK11SlotInfoStr PK11SlotInfo;
typedef struct PK11PreSlotInfoStr PK11PreSlotInfo;
typedef struct PK11SymKeyStr PK11SymKey;
typedef struct PK11ContextStr PK11Context;
typedef struct PK11SlotListStr PK11SlotList;
typedef struct PK11SlotListElementStr PK11SlotListElement;
typedef unsigned long int SECMODModuleID;
typedef struct PK11DefaultArrayEntryStr PK11DefaultArrayEntry;
typedef struct PK11GenericObjectStr PK11GenericObject;
typedef void (*PK11FreeDataFunc) (void *);
typedef enum {
    PK11CertListUnique,
    PK11CertListUser = 1,
    PK11CertListRootUnique = 2,
    PK11CertListCA = 3,
    PK11CertListCAUnique = 4,
    PK11CertListUserUnique = 5,
    PK11CertListAll = 6
} PK11CertListType;
typedef PRUint32 PK11AttrFlags;
typedef enum {
    PK11_OriginNULL,
    PK11_OriginDerive = 1,
    PK11_OriginGenerated = 2,
    PK11_OriginFortezzaHack = 3,
    PK11_OriginUnwrap = 4
} PK11Origin;
typedef enum {
    PK11_DIS_NONE,
    PK11_DIS_USER_SELECTED = 1,
    PK11_DIS_COULD_NOT_INIT_TOKEN = 2,
    PK11_DIS_TOKEN_VERIFY_FAILED = 3,
    PK11_DIS_TOKEN_NOT_PRESENT = 4
} PK11DisableReasons;
typedef enum {
    PK11_TypeGeneric,
    PK11_TypePrivKey = 1,
    PK11_TypePubKey = 2,
typedef char *(*PK11PasswordFunc) (PK11SlotInfo *, PRBool, void *);
typedef struct SECKEYAttributeStr {
    SECItem attrType;
    SECItem **attrValue;
} SECKEYAttribute;
typedef struct SECKEYPrivateKeyInfoStr {
    PLArenaPool *arena;
    SECItem version;
    SECAlgorithmID algorithm;
    SECItem privateKey;
    SECKEYAttribute **attributes;
} SECKEYPrivateKeyInfo;
typedef struct SECKEYEncryptedPrivateKeyInfoStr {
    PLArenaPool *arena;
    SECAlgorithmID algorithm;
    SECItem encryptedData;
    } SECKEYEncryptedPrivateKeyInfo;
typedef enum {
    PK11TokenNotRemovable,
    PK11TokenPresent = 1,
    PK11TokenChanged = 2,
    PK11TokenRemoved = 3
} PK11TokenStatus;
typedef enum {
    PK11TokenRemovedOrChangedEvent,
    PK11TokenPresentEvent = 1
} PK11TokenEvent;

#define _SECOIDT_H_
typedef struct SEC0idDataStr {
    SECItem oid;
    SEC0idTag offset;
    const char *desc;
    unsigned long int mechanism;
    SECSupportExtenTag supportedExtension;
} SEC0idData;
typedef struct SECAlgorithmIDStr {
    SECItem algorithm;
    SECItem parameters;
} SECAlgorithmID;
typedef enum {
    SEC_OID_UNKNOWN,
    SEC_OID_MD2 = 1,
    SEC_OID_MD4 = 2,
    SEC_OID_MD5 = 3,
    SEC_OID_SHA1 = 4,
    SEC_OID_RC2_CBC = 5,
    SEC_OID_RC4 = 6,
    SEC_OID_DES_EDE3_CBC = 7,
    SEC_OID_RC5_CBC_PAD = 8,
    SEC_OID_DES_ECB = 9,
    SEC_OID_DES_CBC = 10,
    SEC_OID_DES_OFB = 11,
    SEC_OID_DES_CFB = 12,
    SEC_OID_DES_MAC = 13,
    SEC_OID_DES_EDE = 14,
    SEC_OID_ISO_SHA_WITH_RSA_SIGNATURE = 15,
    SEC_OID_PKCS1_RSA_ENCRYPTION = 16,
SEC_OID_PKCS1_MD2_WITH_RSA_ENCRYPTION = 17,
SEC_OID_PKCS1_MD4_WITH_RSA_ENCRYPTION = 18,
SEC_OID_PKCS1_MD5_WITH_RSA_ENCRYPTION = 19,
SEC_OID_PKCS1_SHA1_WITH_RSA_ENCRYPTION = 20,
SEC_OID_PKCS5_PBE_WITH_MD2_AND_DES_CBC = 21,
SEC_OID_PKCS5_PBE_WITH_MD5_AND_DES_CBC = 22,
SEC_OID_PKCS5_PBE_WITH_SHA1_AND_DES_CBC = 23,
SEC_OID_PKCS7 = 24,
SEC_OID_PKCS7_DATA = 25,
SEC_OID_PKCS7_SIGNED_DATA = 26,
SEC_OID_PKCS7_ENVELOPED_DATA = 27,
SEC_OID_PKCS7_SIGNED_ENVELOPED_DATA = 28,
SEC_OID_PKCS7_DIGESTED_DATA = 29,
SEC_OID_PKCS7_ENCRYPTED_DATA = 30,
SEC_OID_PKCS9_EMAIL_ADDRESS = 31,
SEC_OID_PKCS9_UNSTRUCTURED_NAME = 32,
SEC_OID_PKCS9_CONTENT_TYPE = 33,
SEC_OID_PKCS9_MESSAGE_DIGEST = 34,
SEC_OID_PKCS9_COUNTER_SIGNATURE = 35,
SEC_OID_PKCS9_CHALLENGE_PASSWORD = 36,
SEC_OID_PKCS9_UNSTRUCTURED_ADDRESS = 38,
SEC_OID_PKCS9_EXTENDED_CERTIFICATE_ATTRIBUTES = 39,
SEC_OID_PKCS9_SMIME_CAPABILITIES = 40,
SEC_OID_AVA_COMMON_NAME = 41,
SEC_OID_AVA_COUNTRY_NAME = 42,
SEC_OID_AVA_LOCALITY = 43,
SEC_OID_AVA_STATE_OR_PROVINCE = 44,
SEC_OID_AVA_ORGANIZATION_NAME = 45,
SEC_OID_AVA_ORGANIZATIONAL_UNIT_NAME = 46,
SEC_OID_AVA_DN_QUALIFIER = 47,
SEC_OID_AVA_DC = 48,
SEC_OID_NS_TYPE_GIF = 49,
SEC_OID_NS_TYPE_JPEG = 50,
SEC_OID_NS_TYPE_URL = 51,
SEC_OID_NS_TYPE_HTML = 52,
SEC_OID_NS_TYPE_CERT_SEQUENCE = 53,
SEC_OID_MISSI_KEA_DSS_OLD = 54,
SEC_OID_MISSI_DSS_OLD = 55,
SEC_OID_MISSI_KEA_DSS = 56,
SEC_OID_MISSI_DSS = 57,
SEC_OID_MISSI_KEA = 58,
SEC_OID_MISSI_ALT_KEA = 59,
SEC_OID_NS_CERT_EXT_NETSCAPE_OK = 60,
SEC_OID_NS_CERT_EXT_ISSUER_LOGO = 61,
SEC_OID_NS_CERT_EXT_SUBJECT_LOGO = 62,
SEC_OID_NS_CERT_EXT_CERT_TYPE = 63,
SEC_OID_NS_CERT_EXT_BASE_URL = 64,
SEC_OID_NS_CERT_EXT_REVOCATION_URL = 65,
SEC_OID_NS_CERT_EXT_CA_REVOCATION_URL = 66,
SEC_OID_NS_CERT_EXT_CA_CRL_URL = 67,
SEC_OID_NS_CERT_EXT_CA_CERT_URL = 68,
SEC_OID_NS_CERT_EXT_CERT_RENEWAL_URL = 69,
SEC_OID_NS_CERT_EXT_CA_POLICY_URL = 70,
SEC_OID_NS_CERT_EXT_HOMEPAGE_URL = 71,
SEC_OID_NS_CERT_EXT_ENTITY_LOGO = 72,
SEC_OID_NS_CERT_EXT_USER_PICTURE = 73,
SEC_OID_NS_CERT_EXT_SSL_SERVER_NAME = 74,
SEC_OID_NS_CERT_EXT_COMMENT = 75,
SEC_OID_NS_CERT_EXT_LOST_PASSWORD_URL = 76,
SEC_OID_NS_CERT_EXT_CERT_RENEWAL_TIME = 77,
SEC_OID_NS_KEY_USAGE_GOVT_APPROVED = 78,
SEC_OID_X509_SUBJECT_DIRECTORY_ATTR = 79,
SEC_OID_X509_SUBJECT_KEY_ID = 80,
SEC_OID_X509_KEY_USAGE = 81,
SEC_OID_X509_PRIVATE_KEY_USAGE_PERIOD = 82,
LSB Core - Generic 5.0

SEC_OID_X509_SUBJECT_ALT_NAME = 83,
SEC_OID_X509_ISSUER_ALT_NAME = 84,
SEC_OID_X509_BASIC_CONSTRAINTS = 85,
SEC_OID_X509_NAME_CONSTRAINTS = 86,
SEC_OID_X509_CRL_DIST_POINTS = 87,
SEC_OID_X509_CERTIFICATE_POLICIES = 88,
SEC_OID_X509_POLICY_MAPPINGS = 89,
SEC_OID_X509_POLICY_CONSTRAINTS = 90,
SEC_OID_X509_AUTH_KEY_ID = 91,
SEC_OID_X509_EXT_KEY_USAGE = 92,
SEC_OID_X509_AUTH_INFO_ACCESS = 93,
SEC_OID_X509_CRL_NUMBER = 94,
SEC_OID_X509_REASON_CODE = 95,
SEC_OID_X509_INVALID_DATE = 96,
SEC_OID_X509_RSA_ENCRYPTION = 97,
SEC_OID_RFC1274_UID = 98,
SEC_OID_RFC1274_MAIL = 99,
SEC_OID_PKCS12 = 100,
SEC_OID_PKCS12_MODE_IDS = 101,
SEC_OID_PKCS12_ESPVK_IDS = 102,
SEC_OID_PKCS12_BAG_IDS = 103,
SEC_OID_PKCS12_CERT_BAG_IDS = 104,
SEC_OID_PKCS12_OIDS = 105,
SEC_OID_PKCS12_PBE_IDS = 106,
SEC_OID_PKCS12_SIGNATURE_IDS = 107,
SEC_OID_PKCS12_ENVELOPING_IDS = 108,
SEC_OID_PKCS12_PKCS8_KEY_SHROUDING = 109,
SEC_OID_PKCS12_KEY_BAG_ID = 110,
SEC_OID_PKCS12_CERT_AND_CRL_BAG_ID = 111,
SEC_OID_PKCS12_SECRET_BAG_ID = 112,
SEC_OID_PKCS12_X509_CERT_CRL_BAG = 113,
SEC_OID_PKCS12_SDSI_CERT_BAG = 114,
SEC_OID_PKCS12_PBE_WITH_SHA1_AND_128_BIT_RC4 = 115,
SEC_OID_PKCS12_PBE_WITH_SHA1_AND_40_BIT_RC4 = 116,
SEC_OID_PKCS12_PBE_WITH_SHA1_AND_TRIPLE_DES_CBC = 117,
SEC_OID_PKCS12_PBE_WITH_SHA1_AND_128_BIT_RC2_CBC = 118,
SEC_OID_PKCS12_PBE_WITH_SHA1_AND_40_BIT_RC2_CBC = 119,
SEC_OID_PKCS12_RSA_ENCRYPTION_WITH_128_BIT_RC4 = 120,
SEC_OID_PKCS12_RSA_ENCRYPTION_WITH_40_BIT_RC4 = 121,
SEC_OID_PKCS12_RSA_ENCRYPTION_WITH_TRIPLE_DES = 122,
SEC_OID_PKCS12_RSA_SIGNATURE_WITH_SHA1_DIGEST = 123,
SEC_OID_ANSIX9_DSA_SIGNATURE = 124,
SEC_OID_ANSIX9_DSA_SIGNATURE_WITH_SHA1_DIGEST = 125,
SEC_OID_BOGUS_DSA_SIGNATURE_WITH_SHA1_DIGEST = 126,
SEC_OID_VERISIGN_USER_NOTICES = 127,
SEC_OID_PKIX_CPS_POINTER_QUALIFIER = 128,
SEC_OID_PKIX_USER_NOTICE_QUALIFIER = 129,
SEC_OID_PKIX_OCSP = 130,
SEC_OID_PKIX_OCSP_BASIC_RESPONSE = 131,
SEC_OID_PKIX_OCSP_NONCE = 132,
SEC_OID_PKIX_OCSP_CRL = 133,
SEC_OID_PKIX_OCSP_RESPONSE = 134,
SEC_OID_PKIX_OCSP_NO_CHECK = 135,
SEC_OID_PKIX_OCSP_ARCHIVE_CUTOFF = 136,
SEC_OID_PKIX_OCSP_SERVICE_LOCATOR = 137,
SEC_OID_PKIX_REGCTRL_REGTOKEN = 138,
SEC_OID_PKIX_REGCTRL_AUTHENTICATOR = 139,
SEC_OID_PKIX_REGCTRL_PKIPUBINFO = 140,
SEC_OID_PKIX_REGCTRL_PKI_ARCH_OPTIONS = 141,
SEC_OID_PKIX_REGCTRL_OLD_CERT_ID = 142,
SEC_OID_PKIX_REGCTRL_PROTOCOL_ENC_KEY = 143,
SEC_OID_PKIX_REGINFO_UTF8_PAIRS = 144,
SEC_OID_PKIX_REGINFO_CERT_REQUEST = 145,
SEC_OID_EXT_KEY_USAGE_SERVER_AUTH = 146,
SEC_OID_EXT_KEY_USAGE_CLIENT_AUTH = 147,
SEC_OID_EXT_KEY_USAGE_CODE_SIGN = 148,
SEC_OID_EXT_KEY_USAGE_EMAIL_PROTECT = 149,
SEC_OID_EXT_KEY_USAGE_TIME_STAMP = 150,
SEC_OID_OCSP_RESPONDER = 151,
SEC_OID_NETSCAPE_SMIME_KEA = 152,
SEC_OID_FORTEZZA_SKIPJACK = 153,
SEC_OID_PKCS12_V2_PBE_WITH_SHA1_AND_128_BIT_RC4 = 154,
SEC_OID_PKCS12_V2_PBE_WITH_SHA1_AND_40_BIT_RC4 = 155,
SEC_OID_PKCS12_V2_PBE_WITH_SHA1_AND_3KEY_TRIPLE_DES_CBC = 156,
SEC_OID_PKCS12_V2_PBE_WITH_SHA1_AND_2KEY_TRIPLE_DES_CBC = 157,
SEC_OID_PKCS12_V2_PBE_WITH_SHA1_AND_128_BIT_RC2_CBC = 158,
SEC_OID_PKCS12_V2_PBE_WITH_SHA1_AND_40_BIT_RC2_CBC = 159,
SEC_OID_PKCS12_SAFE_CONTENTS_ID = 160,
SEC_OID_PKCS12_PKCS8_SHROUDED_KEY_BAG_ID = 161,
SEC_OID_PKCS12_V1_KEY_BAG_ID = 162,
SEC_OID_PKCS12_V1_PKCS8_SHROUDED_KEY_BAG_ID = 163,
SEC_OID_PKCS12_V1_CERT_BAG_ID = 164,
SEC_OID_PKCS12_V1_CRL_BAG_ID = 165,
SEC_OID_PKCS12_V1_SECRET_BAG_ID = 166,
SEC_OID_PKCS12_V1_SAFE_CONTENTS_BAG_ID = 167,
SEC_OID_PKCS9_X509_CERT = 168,
SEC_OID_PKCS9_SDSI_CERT = 169,
SEC_OID_PKCS9_X509_CRL = 170,
SEC_OID_PKCS9_FRIENDLY_NAME = 171,
SEC_OID_PKCS9_LOCAL_KEY_ID = 172,
SEC_OID_BOGUS_KEY_USAGE = 173,
SEC_OID_X942_DIFFIE_HELMAN_KEY = 174,
SEC_OID_NETSCAPE_NICKNAME = 175,
SEC_OID_NETSCAPE_RECOVERY_REQUEST = 176,
SEC_OID_CERT_RENEWAL_LOCATOR = 177,
SEC_OID_NS_CERT_EXT_SCOPE_OF_USE = 178,
SEC_OID_CMS_EPHEMERAL_STATIC_DIFFIE_HELLMAN = 179,
SEC_OID_CMS_3DES_KEY_WRAP = 180,
SEC_OID_CMS_RC2_KEY_WRAP = 181,
SEC_OID_SMIME_ENCRYPTION_KEY_PREFERENCE = 182,
SEC_OID_AES_128_ECB = 183,
SEC_OID_AES_128_CBC = 184,
SEC_OID_AES_192_ECB = 185,
SEC_OID_AES_192_CBC = 186,
SEC_OID_AES_256_ECB = 187,
SEC_OID_AES_256_CBC = 188,
SEC_OID_SDN702_DSA_SIGNATURE = 189,
SEC_OID_MS_SMIME_ENCRYPTION_KEY_PREFERENCE = 190,
SEC_OID_SHA256 = 191,
SEC_OID_SHA384 = 192,
SEC_OID_SHA512 = 193,
SEC_OID_PKCS1_SHA256_WITH_RSA_ENCRYPTION = 194,
SEC_OID_PKCS1_SHA384_WITH_RSA_ENCRYPTION = 195,
SEC_OID_PKCS1_SHA512_WITH_RSA_ENCRYPTION = 196,
SEC_OID_AES_128_KEY_WRAP = 197,
SEC_OID_AES_192_KEY_WRAP = 198,
SEC_OID_AES_256_KEY_WRAP = 199,
SEC_OID_ANSIX962_EC_PUBLIC_KEY = 200,
SEC_OID_ANSIX962_ECDSA_SHA1_SIGNATURE = 201,
SEC_OID_ANSIX962_EC_PRIME192V1 = 202,
SEC_OID_ANSIX962_EC_PRIME192V2 = 203,
SEC_OID_ANSIX962_EC_PRIME192V3 = 204,
SEC_OID_ANSIX962_EC_PRIME239V1 = 205,
SEC_OID_ANSIX962_EC_PRIME239V2 = 206,
SEC_OID_ANSIX962_EC_PRIME239V3 = 207,
SEC_OID_ANSIX962_EC_PRIME256V1 = 208,
SEC_OID_SEQ_EC_SECP112R1 = 209,
SEC_OID_SEQ_EC_SECP112R2 = 210,
SEC_OID_SEQ_EC_SECP128R1 = 211,
SEC_OID_SEQ_EC_SECP128R2 = 212,
SEC_OID_SECG_EC_SECP160K1 = 213,
SEC_OID_SECG_EC_SECP160R1 = 214,
SEC_OID_SECG_EC_SECP160R2 = 215,
SEC_OID_SECG_EC_SECP192K1 = 216,
SEC_OID_SECG_EC_SECP224K1 = 217,
SEC_OID_SECG_EC_SECP224R1 = 218,
SEC_OID_SECG_EC_SECP256K1 = 219,
SEC_OID_SECG_EC_SECP384R1 = 220,
SEC_OID_SECG_EC_SECP521R1 = 221,
SEC_OID_ANSIX962_EC_C2PNB163V1 = 222,
SEC_OID_ANSIX962_EC_C2PNB163V2 = 223,
SEC_OID_ANSIX962_EC_C2PNB163V3 = 224,
SEC_OID_ANSIX962_EC_C2PNB176V1 = 225,
SEC_OID_ANSIX962_EC_C2TNB191V1 = 226,
SEC_OID_ANSIX962_EC_C2TNB191V2 = 227,
SEC_OID_ANSIX962_EC_C2TNB191V3 = 228,
SEC_OID_ANSIX962_EC_C20NB191V4 = 229,
SEC_OID_ANSIX962_EC_C20NB191V5 = 230,
SEC_OID_ANSIX962_EC_C2PNB208W1 = 231,
SEC_OID_ANSIX962_EC_C2TNB239V1 = 232,
SEC_OID_ANSIX962_EC_C2TNB239V2 = 233,
SEC_OID_ANSIX962_EC_C2TNB239V3 = 234,
SEC_OID_ANSIX962_EC_C20NB239V4 = 235,
SEC_OID_ANSIX962_EC_C20NB239V5 = 236,
SEC_OID_ANSIX962_EC_C2PNB272W1 = 237,
SEC_OID_ANSIX962_EC_C2PNB304W1 = 238,
SEC_OID_ANSIX962_EC_C2TNB359V1 = 239,
SEC_OID_ANSIX962_EC_C2PNB368W1 = 240,
SEC_OID_ANSIX962_EC_C2TNB431V1 = 241,
SEC_OID_SECG_EC_SECT113R1 = 242,
SEC_OID_SECG_EC_SECT113R2 = 243,
SEC_OID_SECG_EC_SECT131R1 = 244,
SEC_OID_SECG_EC_SECT131R2 = 245,
SEC_OID_SECG_EC_SECT163K1 = 246,
SEC_OID_SECG_EC_SECT163R1 = 247,
SEC_OID_SECG_EC_SECT163R2 = 248,
SEC_OID_SECG_EC_SECT193R1 = 249,
SEC_OID_SECG_EC_SECT193R2 = 250,
SEC_OID_SECG_EC_SECT233K1 = 251,
SEC_OID_SECG_EC_SECT233R1 = 252,
SEC_OID_SECG_EC_SECT239K1 = 253,
SEC_OID_SECG_EC_SECT283K1 = 254,
SEC_OID_SECG_EC_SECT283R1 = 255,
SEC_OID_SECG_EC_SECT409K1 = 256,
SEC_OID_SECG_EC_SECT409R1 = 257,
SEC_OID_SECG_EC_SECT571K1 = 258,
SEC_OID_SECG_EC_SECT571R1 = 259,
SEC_OID_NETSCAPE_AOLSCREENNAME = 260,
SEC_OID_AVA_SURNAME = 261,
SEC_OID_AVA_SERIAL_NUMBER = 262,
SEC_OID_AVA_STREET_ADDRESS = 263,
SEC_OID_AVA_TITLE = 264,
SEC_OID_AVA_POSTAL_ADDRESS = 265,
SEC_OID_AVA_POSTAL_CODE = 266,
SEC_OID_AVA_POST_OFFICE_BOX = 267,
SEC_OID_AVA_GIVEN_NAME = 268,
SEC_OID_AVA_INITIALIZER = 269,
SEC_OID_AVA_GENERATION_QUALIFIER = 270,
SEC_OID_AVA_HOUSE_IDENTIFIER = 271,
SEC_OID_AVA_PSEUDONYM = 272,
SEC_OID_PKIX_CA_ISSUERS = 273,
SEC_OID_PKCS9_EXTENSION_REQUEST = 274,
SEC_OID_ANSIX962_ECDSA_SIGNATURE_RECOMMENDED_DIGEST = 275,
SEC_OID_ANSIX962_ECDSA_SIGNATURE_SPECIFIED_DIGEST = 276,
SEC_OID_ANSIX962_ECDSA_SHA384_SIGNATURE = 279,
SEC_OID_ANSIX962_ECDSA_SHA512_SIGNATURE = 280,
SEC_OID_X509_HOLD_INSTRUCTION_CODE = 281,
SEC_OID_X509_DELTA_CRL_INDICATOR = 282,
SEC_OID_X509_ISSUING_distribution_POINT = 283,
SEC_OID_X509_CERT_ISSUER = 284,
SEC_OID_X509_FRESHEST_CRL = 285,
SEC_OID_X509_INHIBIT_ANY_POLICY = 286,
SEC_OID_X509_SUBJECT_INFO_ACCESS = 287,
SEC_OID_CAMELLIA_128_CBC = 288,
SEC_OID_CAMELLIA_192_CBC = 289,
SEC_OID_CAMELLIA_256_CBC = 290,
SEC_OID_PKCS5_PBKDF2 = 291,
SEC_OID_PKCS5_PBES2 = 292,
SEC_OID_PKCS5_PBMAC = 293,
SEC_OID_HMAC_SHA1 = 294,
SEC_OID_HMAC_SHA224 = 295,
SEC_OID_HMAC_SHA256 = 296,
SEC_OID_HMAC_SHA384 = 297,
SEC_OID_HMAC_SHA512 = 298,
SEC_OID_PKIX_TIMESTAMPING = 299,
SEC_OID_PKIX_CA_REPOSITORY = 300,
SEC_OID_ISO_SHA1_WITH_RSA_SIGNATURE = 301,
SEC_OID_TOTAL = 302
} SECOidTag;

typedef enum {
    INVALID_CERT_EXTENSION,
    UNSUPPORTED_CERT_EXTENSION = 1,
    SUPPORTED_CERT_EXTENSION = 2
} SECSupportExtenTag;

### 24.4.26 nss3/secpkcs5.h

```c
#define _SECPKCS5_H_

typedef enum {
    pbeBitGenIDNull,
    pbeBitGenCipherKey = 1,
    pbeBitGenCipherIV = 2,
    pbeBitGenIntegrityKey = 3
} PBEBitGenID;

typedef struct PBEBitGenContextStr PBEBitGenContext;
```

### 24.4.27 nss3/secport.h

```c
#define _SECPORT_H_

typedef PRBool("PORTCharConversionWSwapFunc) (PRBool, unsigned
char *,
    unsigned int, unsigned char *,
    unsigned int, unsigned int *,
    unsigned int *,
    PRBool);

typedef PRBool("PORTCharConversionFunc) (PRBool, unsigned char *,
    unsigned int, unsigned
char *,
    unsigned int, unsigned
int *);
```

### 24.5 Interfaces for libssl3

Table 24-5 defines the library name and shared object name for the libssl3 library
The behavior of the interfaces in this library is specified by the following specifications:
- [NSS SSL] Mozilla’s NSS SSL Reference

### 24.5.1 NSS SSL

#### 24.5.1.1 Interfaces for NSS SSL

An LSB conforming implementation shall provide the generic functions for NSS SSL specified in Table 24-6, with the full mandatory functionality as described in the referenced underlying specification.

<table>
<thead>
<tr>
<th>Interface Name</th>
<th>NSS CmpCertChainWC ANames(NSS_3.2)</th>
<th>NSS_FindCertKEAType(NSS_3.2)</th>
<th>NSS_GetClientAuthDataHook(NSS_3.2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSL_ChainVerify</td>
<td>SSL_AuthCertificate(NSS_3.2)</td>
<td>SSL_AuthCertificateHook(NSS_3.2)</td>
<td>SSL_BadCertHook(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_CipherPolicyGet(NSS_3.2)</td>
<td>SSL_CIPHER_PolicyGet(NSS_3.2)</td>
<td>SSL_CIPHER_PolicySet(NSS_3.2)</td>
<td>SSL_CIPHER_PrefGet(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_CipherPrefGetDefault(NSS_3.2)</td>
<td>SSL_CIPHER_PrefSet(NSS_3.2)</td>
<td>SSL_CIPHER_PrefSetDefault(NSS_3.2)</td>
<td>SSL_CIPHER_PrefSetDefault(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_ClearSessionCache(NSS_3.2)</td>
<td>SSL_ClearSessionCache(NSS_3.2)</td>
<td>SSL_ConfigMPServerSIDCache(NSS_3.2)</td>
<td>SSL_CONFIG_SECURE_SERVER(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_ConfigServerSessionCache(NSS_3.2)</td>
<td>SSL_ConfigServerSessionCache(NSS_3.2)</td>
<td>SSL_ConfigServerSessionCache(NSS_3.2)</td>
<td>SSL_ConfigServerSessionCache(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_DataPending(NSS_3.2)</td>
<td>SSL_DataPending(NSS_3.2)</td>
<td>SSL_Dirk(NSS_3.2)</td>
<td>SSL_Dirk(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_ForceHandshake(NSS_3.2)</td>
<td>SSL_ForceHandshake(NSS_3.2)</td>
<td>SSL_GetSessionID(NSS_3.2)</td>
<td>SSL_GetSessionID(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_GetClientAuthDataHook(NSS_3.2)</td>
<td>SSL_GetClientAuthDataHook(NSS_3.2)</td>
<td>SSL_GetClientAuthDataHook(NSS_3.2)</td>
<td>SSL_GetClientAuthDataHook(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_ImportFD(NSS_3.2)</td>
<td>SSL_ImportFD(NSS_3.2)</td>
<td>SSL_InheritMPServerSIDCache(NSS_3.2)</td>
<td>SSL_InvalidateSession(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_OptionsGet(NSS_3.2)</td>
<td>SSL_OptionsGet(NSS_3.2)</td>
<td>SSL_OptionsGetDefault(NSS_3.2)</td>
<td>SSL_OptionsGetDefault(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_OptionsSetDefault(NSS_3.2)</td>
<td>SSL_OptionsSetDefault(NSS_3.2)</td>
<td>SSL_PEER_Certificate(NSS_3.2)</td>
<td>SSL_PEER_Certificate(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_ResetHandshake(NSS_3.2)</td>
<td>SSL_ResetHandshake(NSS_3.2)</td>
<td>SSL_RevealPinArg(NSS_3.2)</td>
<td>SSL_RevealPinArg(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_ResetURL(NSS_3.2)</td>
<td>SSL_ResetURL(NSS_3.2)</td>
<td>SSL_RESET_URL(NSS_3.2)</td>
<td>SSL_RESET_URL(NSS_3.2)</td>
</tr>
</tbody>
</table>

#### 24.6 Data Definitions for libssl3

This section defines global identifiers and their values that are associated with interfaces contained in libssl3. 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 re-
quiring 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.

### 24.6.1 nss3/ecl-exp.h

```c
#define __ecl_exp_h_
#define ECCurve_SECG_CHAR2_163R2        ECCurve_NIST_B163
#define ECCurve_SECG_CHAR2_233R1        ECCurve_NIST_B233
#define ECCurve_WTLS_11 ECCurve_NIST_B233
#define ECCurve_SECG_CHAR2_283R1        ECCurve_NIST_B283
#define ECCurve_SECG_CHAR2_409R1        ECCurve_NIST_B409
#define ECCurve_SECG_CHAR2_571R1        ECCurve_NIST_B571
#define ECCurve_SECG_CHAR2_163K1        ECCurve_NIST_K163
#define ECCurve_WTLS_3 ECCurve_NIST_K163
#define ECCurve_SECG_CHAR2_233K1        ECCurve_NIST_K233
#define ECCurve_WTLS_10 ECCurve_NIST_K233
#define ECCurve_SECG_CHAR2_283K1        ECCurve_NIST_K283
#define ECCurve_SECG_CHAR2_409K1        ECCurve_NIST_K409
#define ECCurve_SECG_PRIME_192R1        ECCurve_NIST_P192
#define ECCurve_X9_62_PRIME_192V1       ECCurve_NIST_P192
#define ECCurve_SECG_PRIME_224R1        ECCurve_NIST_P224
#define ECCurve_WTLS_12 ECCurve_NIST_P224
#define ECCurve_SECG_PRIME_256R1        ECCurve_NIST_P256
#define ECCurve_X9_62_PRIME_256V1       ECCurve_NIST_P256
#define ECCurve_SECG_PRIME_384R1        ECCurve_NIST_P384
#define ECCurve_SECG_PRIME_521R1        ECCurve_NIST_P521
#define ECCurve_WTLS_4 ECCurve_SECG_CHAR2_113R1
#define ECCurve_WTLS_6 ECCurve_SECG_PRIME_112R1
#define ECCurve_WTLS_7 ECCurve_SECG_PRIME_160R1
#define ECCurve_WTLS_5 ECCurve_X9_62_CHAR2_PNB163V1

enum ECField {
    ECField_GFp = 0,
    ECField_GF2m = 1
};
typedef struct ECCurveParamsStr {
    char *text;
    enum ECField field;
    unsigned int size;
    char *irr;
    char *curvea;
    char *curveb;
    char *genx;
    char *geny;
    char *order;
    int cofactor;
} ECCurveParams;
enum ECCurveName {
    ECCurve_noName = 0,
    ECCurve_NIST_P192 = 1,
    ECCurve_NIST_P224 = 2,
    ECCurve_NIST_P256 = 3,
    ECCurve_NIST_P384 = 4,
};
```
ECCurve_NIST_P521 = 5,
ECCurve_NIST_K163 = 6,
ECCurve_NIST_B163 = 7,
ECCurve_NIST_K233 = 8,
ECCurve_NIST_B233 = 9,
ECCurve_NIST_K283 = 10,
ECCurve_NIST_B283 = 11,
ECCurve_NIST_K409 = 12,
ECCurve_NIST_B409 = 13,
ECCurve_NIST_K571 = 14,
ECCurve_NIST_B571 = 15,
ECCurve_X9_62_PRIME_192V2 = 16,
ECCurve_X9_62_PRIME_192V3 = 17,
ECCurve_X9_62_PRIME_239V1 = 18,
ECCurve_X9_62_PRIME_239V2 = 19,
ECCurve_X9_62_PRIME_239V3 = 20,
ECCurve_X9_62_CHAR2_PNB163V1 = 21,
ECCurve_X9_62_CHAR2_PNB163V2 = 22,
ECCurve_X9_62_CHAR2_PNB163V3 = 23,
ECCurve_X9_62_CHAR2_PNB176V1 = 24,
ECCurve_X9_62_CHAR2_PNB191V1 = 25,
ECCurve_X9_62_CHAR2_PNB191V2 = 26,
ECCurve_X9_62_CHAR2_PNB191V3 = 27,
ECCurve_X9_62_CHAR2_PNB208W1 = 28,
ECCurve_X9_62_CHAR2_PNB239V1 = 29,
ECCurve_X9_62_CHAR2_PNB239V2 = 30,
ECCurve_X9_62_CHAR2_PNB239V3 = 31,
ECCurve_X9_62_CHAR2_PNB272W1 = 32,
ECCurve_X9_62_CHAR2_PNB304W1 = 33,
ECCurve_X9_62_CHAR2_PNB304W1 = 34,
ECCurve_X9_62_CHAR2_PNB384W1 = 35,
ECCurve_X9_62_CHAR2_PNB431R1 = 36,
ECCurve_SECG_PRIME_128R1 = 37,
ECCurve_SECG_PRIME_128R2 = 38,
ECCurve_SECG_PRIME_128R3 = 39,
ECCurve_SECG_PRIME_160K1 = 40,
ECCurve_SECG_PRIME_160K2 = 41,
ECCurve_SECG_PRIME_192K1 = 42,
ECCurve_SECG_PRIME_192K2 = 43,
ECCurve_SECG_PRIME_224K1 = 44,
ECCurve_SECG_PRIME_256K1 = 45,
ECCurve_SECG_CHAR2_113R1 = 46,
ECCurve_SECG_CHAR2_113R2 = 47,
ECCurve_SECG_CHAR2_131R1 = 48,
ECCurve_SECG_CHAR2_131R2 = 49,
ECCurve_SECG_CHAR2_131R3 = 50,
ECCurve_SECG_CHAR2_163R1 = 51,
ECCurve_SECG_CHAR2_193R1 = 52,
ECCurve_SECG_CHAR2_193R2 = 53,
ECCurve_SECG_CHAR2_193R3 = 54,
ECCurve_WTLS_1 = 55,
ECCurve_WTLS_8 = 56,
ECCurve_WTLS_9 = 57,
ECCurve_pastLastCurve = 58
};

24.6.2 nss3/ssl.h

#define __ssl_h_
#define SSL_IS_SSL2_CIPHER(which) (((which) & 0xffff) == 0xffff)
#define SSL_REQUIRE_NEVER ((PRBool)0)
#define SSL_REQUIRE_ALWAYS ((PRBool)1)
#define SSL_REQUIRE_FIRST_HANDSHAKE ((PRBool)2)
#define SSL_REQUIRE_NO_ERROR ((PRBool)3)
#define SSL_SECURITY_STATUS_NOOPT       -1
#define SSL_NOT_ALLOWED                  0
#define SSL_SECURITY_STATUS_OFF          0
#define SSL_ALLOWED                       1
#define SSL_SECURITY                      1
#define SSL_SECURITY_STATUS_ON_HIGH      1
#define SSL_REQUIRE_CERTIFICATE          10
#define SSL_ENABLE_FDX                   11
#define SSL_V2_COMPATIBLE_HELLO          12
#define SSL_ENABLE_TLS                    13
#define SSL_ROLLBACK_DETECTION           14
#define SSL_NO_STEP_DOWN                  15
#define SSL_BYPASS_PKCS11                 16
#define SSL_NO_LOCKS                      17
#define SSL_RESTRICTED                    2
#define SSL_SECURITY_STATUS_ON_LOW       2
#define SSL_SOCKS                          2
#define SSL_REQUEST_CERTIFICATE           3
#define SSL_HANDSHAKE_AS_CLIENT           5
#define SSL_HANDSHAKE_AS_SERVER           6
#define SSL_ENABLE_SSL2                   7
#define SSL_ENABLE_SSL3                   8
#define SSL_NO_CACHE                      9
#define SSL_ENV_VAR_NAME                  "SSL_INHERITANCE"

typedef SECStatus(*SSLAuthCertificate) (void *, PRFileDesc *,
PRBool, PRBool);
typedef SECStatus(*SSLGetClientAuthData) (void *, PRFileDesc *,
CERTDistNames *, CERTCertificate *,
SECKEYPrivateKey **);
typedef SECStatus(*SSLBadCertHandler) (void *, PRFileDesc *);
typedef void (*SSLHandshakeCallback) (PRFileDesc *, void *);
extern SECStatus NSS_CmpCertChainWCANames(CERTCertificate * cert,
CERTDistNames * caNames);
extern SSLKEAType NSS_FindCertKEAType(CERTCertificate * cert);
extern SECStatus NSS_GetClientAuthData(void *arg, PRFileDesc *,
socket,
struct CERTDistNamesStr *caNames,
struct CERTCertificateStr **pRetCert,
struct SECKEYPrivateKeyStr **pRetKey);
extern SECStatus SSL_AuthCertificate(void *arg, PRFileDesc * fd,
PRBool checkSig, PRBool isServer);
extern SECStatus SSL_AuthCertificateHook(PRFileDesc * fd,
SSLAuthCertificate f,
void *arg);
extern SECStatus SSL_BadCertHook(PRFileDesc * fd,
SSLBadCertHandler f,
void *arg);
extern SECStatus SSL_CIPHERPolicyGet(PRInt32 cipher, PRInt32 *
policy);
extern SECStatus SSL_CIPHERPolicySet(PRInt32 cipher, PRInt32 *
policy);
extern SECStatus SSL_CIPHERPrefGet(PRFileDesc * fd, PRInt32 cipher,
PRBool * enabled);
extern SECStatus SSL_CIPHERPrefGetDefault(PRInt32 cipher,
PRBool * enabled);
extern SECStatus SSL_CIPHERPrefSet(PRFileDesc * fd, PRInt32 cipher,
#include <secitem.h>
#include <secstatus.h>
#include <sslerr.h>
#include <sys/prtime.h>

extern void SSL_PurgeCache(void);

extern SECStatus SSL_ConfigMPServerSIDCache(int maxCacheEntries,
                                           PRUint32 timeout,
                                           PRUint32
                                           *directory);

extern SECStatus SSL_ConfigSecureServer(PRFileDesc * fd, CERTCertificate * cert,
                                        SECKEYPrivateKey * key,
                                        SSLKEAType kea);

extern SECStatus SSL_ConfigServerSessionIDCache(int maxCacheEntries,
                                                  PRUint32 timeout,
                                                  PRUint32
                                                  *directory);

extern int SSL_DataPending(PRFileDesc * fd);

extern SECStatus SSL_ForceHandshake(PRFileDesc * fd);

extern SECStatus SSL_GetClientAuthDataHook(PRFileDesc * fd,
                                           SSLGetClientAuthData
                                           * directory);

extern SECItem *SSL_GetSessionID(PRFileDesc * fd);

extern SECStatus SSL_HandshakeCallback(PRFileDesc * fd,
                                        SSLHandshakeCallback cb,
                                        void * client_data);

extern PRFileDesc *SSL_ImportFD(PRFileDesc * model, PRFileDesc * fd);

extern SECStatus SSL_InheritMPServerSIDCache(const char
                                             *envString);

extern SECStatus SSL_InvalidateSession(PRFileDesc * fd);

extern SECStatus SSL_OptionGet(PRFileDesc * fd, PRInt32 option,
                               PRBool * on);

extern SECStatus SSL_OptionGetDefault(PRInt32 option, PRBool *
                                       on);

extern SECStatus SSL_OptionSet(PRFileDesc * fd, PRInt32 option,
                               PRBool on);

extern SECStatus SSL_OptionSetDefault(PRInt32 option, PRBool on);

extern CERTCertificate *SSL_PeerCertificate(PRFileDesc * fd);

extern SECStatus SSL_ReHandshake(PRFileDesc * fd, PRBool
                                   flushCache);

extern SECStatus SSL_ResetHandshake(PRFileDesc * fd, PRBool
                                     asServer);

extern void *SSL_RevealPinArg(PRFileDesc * socket);

extern char *SSL_RevealURL(PRFileDesc * socket);

extern SECStatus SSL_SecurityStatus(PRFileDesc * fd, int *on,
                                    char **cipher, int *keySize,
                                    int *secretKeySize, char
                                    **issuer,
                                    char **subject);

extern SECStatus SSL_SetPKCS11PinArg(PRFileDesc * fd, void *a);

extern SECStatus SSL_SetSockPeerID(PRFileDesc * fd, const char
                                   *peerID);

extern SECStatus SSL_SetURL(PRFileDesc * fd, const char *url);

24.6.3 nss3/sslerr.h

#define __SSL_ERR_H_
#define IS_SSL_ERROR(code)       
    (((code) >= SSL_ERROR_BASE) && ((code) <
```c
typedef enum {
    SSL_ERROR_EXPORT_ONLY_SERVER = (SSL_ERROR_BASE + 0),
    SSL_ERROR_US_ONLY_SERVER = (SSL_ERROR_BASE + 1),
    SSL_ERROR_NO_CYPHER_OVERLAP = (SSL_ERROR_BASE + 2),
    SSL_ERROR_NO_CERTIFICATE = (SSL_ERROR_BASE + 3),
    SSL_ERROR_BAD_CERTIFICATE = (SSL_ERROR_BASE + 4),
    SSL_ERROR_NO_CYPHER_OVERLAP = (SSL_ERROR_BASE + 5),
    SSL_ERROR_BAD_CLIENT = (SSL_ERROR_BASE + 6),
    SSL_ERROR_BAD_SERVER = (SSL_ERROR_BASE + 7),
    SSL_ERROR_UNSUPPORTED_CERTIFICATE_TYPE = (SSL_ERROR_BASE + 8),
    SSL_ERROR_UNSUPPORTED_VERSION = (SSL_ERROR_BASE + 9),
    SSL_ERROR_Wrong_CERTIFICATE = (SSL_ERROR_BASE + 11),
    SSL_ERROR_BAD_CERT_DOMAIN = (SSL_ERROR_BASE + 12),
    SSL_ERROR_POST_WARNING = (SSL_ERROR_BASE + 13),
    SSL_ERROR_SSL2_DISABLED = (SSL_ERROR_BASE + 14),
    SSL_ERROR_BAD_MAC_READ = (SSL_ERROR_BASE + 15),
    SSL_ERROR_BAD_MAC_ALERT = (SSL_ERROR_BASE + 16),
    SSL_ERROR_BAD_ALERT = (SSL_ERROR_BASE + 17),
    SSL_ERROR_EXPRESSED_CERT_ALERT = (SSL_ERROR_BASE + 18),
    SSL_ERROR_SSL_DISABLED = (SSL_ERROR_BASE + 19),
    SSL_ERROR_SSL_DISABLED = (SSL_ERROR_BASE + 20),
    SSL_ERROR_Fortezza_PQG = (SSL_ERROR_BASE + 21),
    SSL_ERROR_UNKNOWN_CIPHER_SUITE = (SSL_ERROR_BASE + 22),
    SSL_ERROR_NO_CIPHERS_SUPPORTED = (SSL_ERROR_BASE + 23),
    SSL_ERROR_BAD_BLOCK_PADDING = (SSL_ERROR_BASE + 24),
    SSL_ERROR_RX_RECORD_TOO_LONG = (SSL_ERROR_BASE + 25),
    SSL_ERROR_TX_RECORD_TOO_LONG = (SSL_ERROR_BASE + 26),
    SSL_ERROR_RX_MALFORMED_CLIENT_HELLO = (SSL_ERROR_BASE + 27),
    SSL_ERROR_RX_MALFORMED_CERT_REQUEST = (SSL_ERROR_BASE + 28),
    SSL_ERROR_RX_MALFORMED_CERTIFICATE = (SSL_ERROR_BASE + 29),
    SSL_ERROR_RX_MALFORMED_CLIENT_KEY_EXCH = (SSL_ERROR_BASE + 30),
    SSL_ERROR_RX_MALFORMED_SERVER_KEY_EXCH = (SSL_ERROR_BASE + 31),
    SSL_ERROR_RX_MALFORMED_CERT_REQUEST = (SSL_ERROR_BASE + 32),
    SSL_ERROR_RX_MALFORMED_HELLO_DONE = (SSL_ERROR_BASE + 33),
    SSL_ERROR_RX_MALFORMED_CERT_VERIFY = (SSL_ERROR_BASE + 34),
    SSL_ERROR_RX_MALFORMED_CLIENT_KEY_EXCH = (SSL_ERROR_BASE + 35),
    SSL_ERROR_RX_MALFORMED_FINISHED = (SSL_ERROR_BASE + 36),
    SSL_ERROR_RX_MALFORMED_CHANGE_CIPHER = (SSL_ERROR_BASE + 37),
    SSL_ERROR_RX_MALFORMED_ALERT = (SSL_ERROR_BASE + 38),
    SSL_ERROR_RX_MALFORMED_HANDSHAKE = (SSL_ERROR_BASE + 39),
    SSL_ERROR_RX_MALFORMED_APPLICATION_DATA = (SSL_ERROR_BASE + 40),
    SSL_ERROR_RX_UNEXPECTED_HELLO_REQUEST = (SSL_ERROR_BASE + 41),
    SSL_ERROR_RX_UNEXPECTED_CLIENT_HELLO = (SSL_ERROR_BASE + 42),
    SSL_ERROR_RX_UNEXPECTED_SERVER_HELLO = (SSL_ERROR_BASE + 43),
    SSL_ERROR_RX_UNEXPECTED_CERTIFICATE = (SSL_ERROR_BASE + 44),
    SSL_ERROR_RX_UNEXPECTED_SERVER_KEY_EXCH = (SSL_ERROR_BASE + 45),
    SSL_ERROR_RX_UNEXPECTED_CERT_REQUEST = (SSL_ERROR_BASE + 46),
    SSL_ERROR_RX_UNEXPECTED_HELLO_DONE = (SSL_ERROR_BASE + 47),
    SSL_ERROR_RX_UNEXPECTED_CERT_VERIFY = (SSL_ERROR_BASE + 48),
    SSL_ERROR_RX_UNEXPECTED_CLIENT_KEY_EXCH = (SSL_ERROR_BASE + 49),
    SSL_ERROR_RX_UNEXPECTED_FINISHED = (SSL_ERROR_BASE + 50),
    SSL_ERROR_RX_UNEXPECTED_CHANGE_CIPHER = (SSL_ERROR_BASE + 51),
    SSL_ERROR_RX_UNEXPECTED_ALERT = (SSL_ERROR_BASE + 52),
    SSL_ERROR_RX_UNEXPECTED_HANDSHAKE = (SSL_ERROR_BASE + 53),
    SSL_ERROR_RX_UNEXPECTED_APPLICATION_DATA = (SSL_ERROR_BASE +

```
SSL_ERROR_RX_UNKNOWN_RECORD_TYPE = (SSL_ERROR_BASE + 55),
SSL_ERROR_RX_UNKNOWN_HANDSHAKE = (SSL_ERROR_BASE + 56),
SSL_ERROR_RX_UNKNOWN_ALERT = (SSL_ERROR_BASE + 57),
SSL_ERROR_CLOSE_NOTIFY_ALERT = (SSL_ERROR_BASE + 58),
SSL_ERROR_HANDSHAKE_UNEXPECTED_ALERT = (SSL_ERROR_BASE + 59),
SSL_ERROR_DECOMPRESSION_FAILURE_ALERT = (SSL_ERROR_BASE + 60),
SSL_ERROR_HANDSHAKE_FAILURE_ALERT = (SSL_ERROR_BASE + 61),
SSL_ERROR_ILLEGAL_PARAMETER_ALERT = (SSL_ERROR_BASE + 62),
SSL_ERROR_UNSUPPORTED_CERT_ALERT = (SSL_ERROR_BASE + 63),
SSL_ERROR_CERTIFICATE_UNKNOWN_ALERT = (SSL_ERROR_BASE + 64),
SSL_ERROR_SIGN_HASHES_FAILURE = (SSL_ERROR_BASE + 65),
SSL_ERROR_EXTRACT_PUBLIC_KEY_FAILURE = (SSL_ERROR_BASE + 66),
SSL_ERROR_CLIENT_KEY_EXCHANGE_FAILURE = (SSL_ERROR_BASE + 67),
SSL_ERROR_CPU_ARITHMETIC_FAILURE = (SSL_ERROR_BASE + 68),
SSL_ERROR_CLIENT_KEY_EXCHANGE_FAILURE = (SSL_ERROR_BASE + 69),
SSL_ERROR_ENCRYPTION_FAILURE = (SSL_ERROR_BASE + 70),
SSL_ERROR_DECRYPTION_FAILURE = (SSL_ERROR_BASE + 71),
SSL_ERROR_SOCKET_WRITE_FAILURE = (SSL_ERROR_BASE + 72),
SSL_ERROR_MD5_DIGEST_FAILURE = (SSL_ERROR_BASE + 73),
SSL_ERROR_SHA_DIGEST_FAILURE = (SSL_ERROR_BASE + 74),
SSL_ERROR_MAC_COMPUTATION_FAILURE = (SSL_ERROR_BASE + 75),
SSL_ERROR_SYM_KEY_CONTEXT_FAILURE = (SSL_ERROR_BASE + 76),
SSL_ERROR_SYM_KEY_UNWRAP_FAILURE = (SSL_ERROR_BASE + 77),
SSL_ERROR_PUB_KEY_SIZE_LIMIT_EXCEEDED = (SSL_ERROR_BASE + 78),
SSL_ERROR_IV_PARAM_FAILURE = (SSL_ERROR_BASE + 79),
SSL_ERROR_INIT_CIPHER_SUITE_FAILURE = (SSL_ERROR_BASE + 80),
SSL_ERROR_SESSION_KEY_GEN_FAILURE = (SSL_ERROR_BASE + 81),
SSL_ERROR_NO_SERVER_KEY_FOR_ALG = (SSL_ERROR_BASE + 82),
SSL_ERROR_TOKEN_INSERTION_REMOVAL = (SSL_ERROR_BASE + 83),
SSL_ERROR_TOKEN_SLOT_NOT_FOUND = (SSL_ERROR_BASE + 84),
SSL_ERROR_COMPRESS_NOT_CONFIGURED = (SSL_ERROR_BASE + 85),
SSL_ERROR_HANDSHAKE_NOT_COMPLETED = (SSL_ERROR_BASE + 86),
SSL_ERROR_HANDSHAKE_HASH_OVERFLOW = (SSL_ERROR_BASE + 87),
SSL_ERROR_CERTIFICATE_UNOBTAINABLE_ALERT = (SSL_ERROR_BASE + 88),
SSL_ERROR_DECRYPT_HASH_VALUE_ALERT = (SSL_ERROR_BASE + 89),
SSL_ERROR_SESSION_NOT_FOUND = (SSL_ERROR_BASE + 90),
SSL_ERROR_CERTIFICATE_INVALID_ALERT = (SSL_ERROR_BASE + 91),
SSL_ERROR_SESSION_KEY_GEN_FAILURE = (SSL_ERROR_BASE + 92),
SSL_ERROR_CLOCK_MISMATCH_ALERT = (SSL_ERROR_BASE + 93),
SSL_ERROR_ACCESS_DENIED_ALERT = (SSL_ERROR_BASE + 94),
SSL_ERROR_DECRYPT_ERROR_ALERT = (SSL_ERROR_BASE + 95),
SSL_ERROR_EXPORT_RESTRICTION_ALERT = (SSL_ERROR_BASE + 96),
SSL_ERROR_PROTOCOL_VERSION_ALERT = (SSL_ERROR_BASE + 97),
SSL_ERROR_INSUFFICIENT_SECURITY_ALERT = (SSL_ERROR_BASE + 98),
SSL_ERROR_INTERNAL_ERROR_ALERT = (SSL_ERROR_BASE + 99),
SSL_ERROR_USER_AUTHENTICATION_ALERT = (SSL_ERROR_BASE + 100),
SSL_ERROR_USER_CANCELED_ALERT = (SSL_ERROR_BASE + 101),
SSL_ERROR_NO_RENEGOTIATION_ALERT = (SSL_ERROR_BASE + 102),
SSL_ERROR_SERVER_CACHE_NOT_CONFIGURED = (SSL_ERROR_BASE + 103),
SSL_ERROR_UNSUPPORTED_EXTENSION_ALERT = (SSL_ERROR_BASE + 104),
SSL_ERROR_UNSUPPORTED_CERTIFICATE_SUPPORT = (SSL_ERROR_BASE + 105),
SSL_ERROR_UI_REMAINED_NAME_ALERT = (SSL_ERROR_BASE + 106),
SSL_ERROR_BAD_CERT_STATUS_RESPONSE_ALERT = (SSL_ERROR_BASE + 107),
SSL_ERROR_BAD_CERT_HASH_VALUE_ALERT = (SSL_ERROR_BASE + 108) } SSLLErrorCodes;
LSB Core - Generic 5.0

24 Libraries

24.6.4 nss3/sslproto.h
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define
#define

__sslproto_h_
SSL_MT_ERROR
0
SSL_NULL_WITH_NULL_NULL 0x0000
SSL_PE_NO_CYPHERS
0x0001
SSL_RSA_WITH_NULL_MD5
0x0001
SSL_LIBRARY_VERSION_2
0x0002
SSL_PE_NO_CERTIFICATE
0x0002
SSL_RSA_WITH_NULL_SHA
0x0002
SSL_RSA_EXPORT_WITH_RC4_40_MD5 0x0003
SSL_PE_BAD_CERTIFICATE 0x0004
SSL_RSA_WITH_RC4_128_MD5
0x0004
SSL_RSA_WITH_RC4_128_SHA
0x0005
SSL_PE_UNSUPPORTED_CERTIFICATE_TYPE
SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5
SSL_RSA_WITH_IDEA_CBC_SHA
0x0007
SSL_RSA_EXPORT_WITH_DES40_CBC_SHA
SSL_RSA_WITH_DES_CBC_SHA
0x0009
SSL_RSA_WITH_3DES_EDE_CBC_SHA
0x000a
SSL_DH_DSS_EXPORT_WITH_DES40_CBC_SHA
SSL_DH_DSS_WITH_DES_CBC_SHA
0x000c
SSL_DH_DSS_WITH_3DES_EDE_CBC_SHA
SSL_DH_RSA_EXPORT_WITH_DES40_CBC_SHA
SSL_DH_RSA_WITH_DES_CBC_SHA
0x000f
SSL_DH_RSA_WITH_3DES_EDE_CBC_SHA
SSL_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA
SSL_DHE_DSS_WITH_DES_CBC_SHA
0x0012
SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA
SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA
SSL_DHE_RSA_WITH_DES_CBC_SHA
0x0015
SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA
SSL_DH_ANON_EXPORT_WITH_RC4_40_MD5
SSL_DH_ANON_WITH_RC4_128_MD5
0x0018
SSL_DH_ANON_EXPORT_WITH_DES40_CBC_SHA
SSL_DH_ANON_WITH_DES_CBC_SHA
0x001a
SSL_DH_ANON_WITH_3DES_EDE_CBC_SHA
SSL_FORTEZZA_DMS_WITH_NULL_SHA 0x001c
SSL_FORTEZZA_DMS_WITH_FORTEZZA_CBC_SHA
SSL_FORTEZZA_DMS_WITH_RC4_128_SHA
TLS_RSA_WITH_AES_128_CBC_SHA
0x002F
TLS_DH_DSS_WITH_AES_128_CBC_SHA 0x0030
TLS_DH_RSA_WITH_AES_128_CBC_SHA 0x0031
TLS_DHE_DSS_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_DH_ANON_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
0x0035
TLS_DH_DSS_WITH_AES_256_CBC_SHA 0x0036
TLS_DH_RSA_WITH_AES_256_CBC_SHA 0x0037
TLS_DHE_DSS_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA
TLS_DH_ANON_WITH_AES_256_CBC_SHA
TLS_RSA_EXPORT1024_WITH_DES_CBC_SHA
TLS_DHE_DSS_EXPORT1024_WITH_DES_CBC_SHA
TLS_RSA_EXPORT1024_WITH_RC4_56_SHA
TLS_DHE_DSS_EXPORT1024_WITH_RC4_56_SHA
TLS_DHE_DSS_WITH_RC4_128_SHA
0x0066
SSL_AT_MD5_WITH_RSA_ENCRYPTION 0x01
SSL_CK_RC4_128_WITH_MD5 0x01
SSL_CT_X509_CERTIFICATE 0x01
SSL_CK_RC4_128_EXPORT40_WITH_MD5
SSL_CK_RC2_128_CBC_WITH_MD5
0x03
SSL_LIBRARY_VERSION_3_0 0x0300
SSL_LIBRARY_VERSION_3_1_TLS
0x0301
SSL_CK_RC2_128_CBC_EXPORT40_WITH_MD5

© 2015 Linux Foundation

0x0006
0x0006
0x0008
0x000b
0x000d
0x000e
0x0010
0x0011
0x0013
0x0014
0x0016
0x0017
0x0019
0x001b
0x001d
0x001e

0x0032
0x0033
0x0034

0x0038
0x0039
0x003A
0x0062
0x0063
0x0064
0x0065

0x02

0x04

907


#define SSL_CK_IDEA_128_CBC_WITH_MD5 0x05
#define SSL_CK_DES_64_CBC_WITH_MD5 0x06
#define SSL_CK_DES_128_CBC_WITH_MD5 0x07
#define TLS_ECDH_ECDSA_WITH_NULL_SHA 0xC001
#define TLS_ECDH_ECDSA_WITH_RC4_128_SHA 0xC002
#define TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA 0xC003
#define TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA 0xC004
#define TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA 0xC005
#define TLS_ECDHE_ECDSA_WITH_NULL_SHA 0xC006
#define TLS_ECDHE_ECDSA_WITH_RC4_128_SHA 0xC007
#define TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA 0xC008
#define TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA 0xC009
#define TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA 0xC00A
#define TLS_ECDH_RSA_WITH_NULL_SHA 0xC00B
#define TLS_ECDH_RSA_WITH_RC4_128_SHA 0xC00C
#define TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA 0xC00D
#define TLS_ECDH_RSA_WITH_AES_128_CBC_SHA 0xC00E
#define TLS_ECDH_RSA_WITH_AES_256_CBC_SHA 0xC00F
#define TLS_ECDH_rsa_with_null_sha 0xC010
#define TLS_ECDH_RSA_WITH_RC4_128_SHA 0xC011
#define TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA 0xC012
#define TLS_ECDH_RSA_WITH_AES_128_CBC_SHA 0xC013
#define TLS_ECDH_anon_NULL_SHA 0xC014
#define TLS_ECDH_anon_WITH_NULL_SHA 0xC015
#define TLS_ECDH_anon_WITH_RC4_128_SHA 0xC016
#define TLS_ECDH_anon_WITH_AES_128_CBC_SHA 0xC017
#define TLS_ECDH_anon_WITH_AES_256_CBC_SHA 0xC018
#define SSL_RSA_FIPS_WITH_DES_CBC_SHA 0xfefe
#define SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA 0xfff
#define SSL_RSA_OLDFIPS_WITH_3DES_EDE_CBC_SHA 0xffe0
#define SSL_RSA_OLDFIPS_WITH_DES_CBC_SHA 0xffe1
#define SSL_HL_CLIENT_FINISHED_HBYTES 1
#define SSL_HL_SERVER_FINISHED_HBYTES 1
#define SSL_HL_SERVER_VERIFY_HBYTES 1
#define SSL_MT_CLIENT_HELLO 1
#define SSL_HL_CLIENT_MASTER_KEY_HBYTES 10
#define SSL_HL_CLIENT_MASTER_KEY_HBYTES 11
#define SSL_HL_REQUEST_CERTIFICATE_HBYTES 2
#define SSL_MT_CLIENT_MASTER_KEY 2
#define SSL_HL_ERROR_HBYTES 3
#define SSL_MT_CLIENT_FINISHED 3
#define SSL_MT_SERVER_HELLO 4
#define SSL_MT_SERVER_VERIFY 5
#define SSL_HL_CLIENT_CERTIFICATE_HBYTES 6
#define SSL_HL_CLIENT_FINISH_HBYTES 6
#define SSL_HL_CLIENT_CERTIFICATE_HBYTES 7
#define SSL_MT_CLIENT_REQUEST 8
#define SSL_MT_CLIENT_HELLO_HBYTES 9

24.6.5 nss3/sslt.h

#define __sslt_h_

typedef enum {
    ssl_kea_null,
    ssl_kea_rsa = 1,
    ssl_kea_dh = 2,
    ssl_kea_fortezza = 3,
    ssl_kea_ecdh = 4,
    ssl_kea_size = 5
} SSLKEAType;
typedef enum {
    ssl_sign_null,
    ssl_sign_rsa = 1,
typedef enum {
  ssl_sign_null,
  ssl_sign_dsa = 2,
  ssl_sign_ecdsa = 3
} SSLSignType;

typedef enum {
  ssl_auth_null,
  ssl_auth_rsa = 1,
  ssl_auth_dsa = 2,
  ssl_auth_kea = 3,
  ssl_auth_ecdsa = 4
} SSLAuthType;

typedef enum {
  ssl_calg_null,
  ssl_calg_rc4 = 1,
  ssl_calg_rc2 = 2,
  ssl_calg_des = 3,
  ssl_calg_3des = 4,
  ssl_calg IDEA = 5,
  ssl_calg_fortezza = 6,
  ssl_calg_aes = 7,
  ssl_calg_camellia = 8
} SSLCipherAlgorithm;

typedef enum {
  ssl_mac_null,
  ssl_mac_md5 = 1,
  ssl_mac_sha = 2,
  ssl_hmac_md5 = 3,
  ssl_hmac_sha = 4
} SSLMACAlgorithm;

typedef struct SSLChannelInfoStr {
  PRUint32 length;
  PRUint16 protocolVersion;
  PRUint16 cipherSuite;
  PRUint32 authKeyBits;
  PRUint32 keaKeyBits;
  PRUint32 creationTime;
  PRUint32 lastAccessTime;
  PRUint32 expirationTime;
  PRUint32 sessionIDLength;
  PRUint8 sessionID[31];
} SSLChannelInfo;

typedef struct SSLCipherSuiteInfoStr {
  PRUint16 length;
  PRUint16 cipherSuite;
  const char *cipherSuiteName;
  const char *authAlgorithmName;
  SSLAuthType authAlgorithm;
  const char *keaTypeName;
  SSLKEAType keaType;
  const char *symCipherName;
  SSLCipherAlgorithm symCipher;
  PRUint16 symKeyBits;
  PRUint16 symKeySpace;
  PRUint16 effectiveKeyBits;
  const char *macAlgorithmName;
  SSLMACAlgorithm macAlgorithm;
  PRUint16 macBits;
  PRUintn isFIPS:1;
  PRUintn isExportable:1;
  PRUintn nonStandard:1;
  PRUintn reservedBits:29;
} SSLCipherSuiteInfo;
XI Package Format and Installation
25 Software Installation

25.1 Introduction

Applications shall either be packaged in the RPM packaging format as defined in this specification, or supply an installer which is LSB conforming (for example, calls LSB commands and utilities).

Note: Supplying an RPM format package is encouraged because it makes systems easier to manage. This specification does not require the implementation to use RPM as the package manager; it only specifies the format of the package file and requires that implementations must have some method of installing conforming packages.

Applications are also encouraged to uninstall cleanly.

A package in the RPM format may include a dependency on the LSB Core and other LSB specifications, as described in Section 25.6. Packages that are not in the RPM format may test for the presence of a conforming implementation by means of the lsb_release utility.

Implementations shall provide a mechanism for installing applications in the RPM packaging format with some restrictions listed below.

Note: The implementation itself may use a different packaging format for its own packages, and may use any available mechanism for installing conforming packages, including translation into a different format.

25.2 Package File Format

An RPM format file consists of 4 sections, the Lead, Signature, Header, and the Payload. All values are stored in network byte order.

Table 25-1 RPM File Format

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lead</td>
<td>Used to identify the package file.</td>
</tr>
<tr>
<td>Signature</td>
<td>Used to verify the integrity, and optionally, the authenticity of the majority of the package file.</td>
</tr>
<tr>
<td>Header</td>
<td>Contains all available information about the package. Entries such as the package's name, version, and file list, are contained in the header.</td>
</tr>
<tr>
<td>Payload</td>
<td>Holds the files to be installed.</td>
</tr>
</tbody>
</table>

These 4 sections shall exist in the order specified.

The lead section is used to identify the package file.

The signature section is used to verify the integrity, and optionally, the authenticity of the majority of the package file.

The header section contains all available information about the package. Entries such as the package's name, version, and file list, are contained in the header.

The payload section holds the files to be installed.

25.2.1 Lead Section

```c
struct rpmlead {
    unsigned char magic[4];
    unsigned char major, minor;
    short type;
    short archnum;
    char name[66];
    short osnum;
    short signature_type;
    char reserved[16];
};
```
magic
Value identifying this file as an RPM format file. This value shall be "\355\253\356\333".

major
Value indicating the major version number of the file format version. This value shall be 3.

minor
Value indicating the minor revision number of file format version. This value shall be 0.

type
Value indicating whether this is a source or binary package. This value shall be 0 to indicate a binary package.

archnum
Value indicating the architecture for which this package is valid. This value is specified in the relevant architecture specific part of the LSB Core Specification.

name
A NUL terminated string that provides the package name. This name shall conform with the Package Naming Conventions section of this specification.

osnum
Value indicating the Operating System for which this package is valid. This value shall be 1.

signature_type
Value indicating the type of the signature used in the Signature part of the file. This value shall be 5.

reserved
Reserved space. The value is undefined.

25.2.2 Header Structure

The Header structure is used for both the Signature and Header Sections. A Header Structure consists of 3 parts, a Header record, followed by 1 or more Index records, followed by 0 or more bytes of data associated with the Index records. A Header structure shall be aligned to an 8 byte boundary.

Table 25-2 Signature Format

<table>
<thead>
<tr>
<th>Header Record</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array of Index Records</td>
</tr>
<tr>
<td>Store of Index Values</td>
</tr>
</tbody>
</table>

25.2.2.1 Header Record

struct rpmheader {
    unsigned char magic[4];
    unsigned char reserved[4];
    int nindex;
    int hsize;
};
LSB Core - Generic 5.0

**magic**

Value identifying this record as an RPM header record. This value shall be "\216\255\350\001".

**reserved**

Reserved space. This value shall be "\000\000\000\000".

**nindex**

The number of Index Records that follow this Header Record. There should be at least 1 Index Record.

**hsize**

The size in bytes of the storage area for the data pointed to by the Index Records.

### 25.2.2.2 Index Record

```c
struct rpmhdrindex {
    int tag;
    int type;
    int offset;
    int count;
};
```

**tag**

Value identifying the purpose of the data associated with this Index Record. The value of this field is dependent on the context in which the Index Record is used, and is defined below and in later sections.

**type**

Value identifying the type of the data associated with this Index Record. The possible `type` values are defined below.

**offset**

Location in the Store of the data associated with this Index Record. This value should between 0 and the value contained in the `hsize` of the Header Structure.

**count**

Size of the data associated with this Index Record. The `count` is the number of elements whose size is defined by the type of this Record.

### 25.2.2.2.1 Index Type Values

The possible values for the `type` field are defined in this table.

<table>
<thead>
<tr>
<th>Type</th>
<th>Value</th>
<th>Size (in bytes)</th>
<th>Alignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPM_NULL_TYPE</td>
<td>0</td>
<td>Not Implemented.</td>
<td></td>
</tr>
<tr>
<td>RPM_CHAR_TYPE</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>RPM_INT8_TYPE</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>RPM_INT16_TYPE</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>RPM_INT32_TYPE</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>RPM_INT64_TYPE</td>
<td>5</td>
<td>Reserved.</td>
<td></td>
</tr>
<tr>
<td>RPM_STRING_TYPE</td>
<td>6</td>
<td>variable, NUL terminated</td>
<td>1</td>
</tr>
</tbody>
</table>
### Table 25-4 Header Private Tag Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_HEADERSIGNATURES</td>
<td>62</td>
<td>BIN</td>
<td>16</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_HEADERIMMUTABLE</td>
<td>63</td>
<td>BIN</td>
<td>16</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_HEADERI18NTABLE</td>
<td>100</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
</tbody>
</table>

**RPMTAG_HEADERSIGNATURES**

The signature tag differentiates a signature header from a metadata header, and identifies the original contents of the signature header.

**RPMTAG_HEADERIMMUTABLE**

This tag contains an index record which specifies the portion of the Header Record which was used for the calculation of a signature. This data shall be preserved or any header-only signature will be invalidated.

**RPMTAG_HEADERI18NTABLE**

Contains a list of locales for which strings are provided in other parts of the package.

Not all Index records defined here will be present in all packages. Each tag value has a status which is defined here.

**Required**

This Index Record shall be present.

**Optional**

This Index Record may be present.
Informational
This Index Record may be present, but does not contribute to the processing of the package.

Deprecated
This Index Record should not be present.

Obsolete
This Index Record shall not be present.

Reserved
This Index Record shall not be present.

25.2.2.3 Header Store
The header store contains the values specified by the Index structures. These values are aligned according to their type and padding is used if needed. The store is located immediately following the Index structures.

25.2.3 Signature Section
The Signature section is implemented using the Header structure. The signature section defines the following additional tag values which may be used in the Index structures.
These values exist to provide additional information about the rest of the package.

<table>
<thead>
<tr>
<th>Table 25-5 Signature Tag Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>------------------------</td>
</tr>
<tr>
<td>RPMSIGTAG_SIZE</td>
</tr>
<tr>
<td>RPMSIGTAG_PAYLOADSIZE</td>
</tr>
</tbody>
</table>

RPMSIGTAG_SIZE
This tag specifies the combined size of the Header and Payload sections.

RPMSIGTAG_PAYLOADSIZE
This tag specifies the uncompressed size of the Payload archive, including the cpio headers.
These values exist to ensure the integrity of the rest of the package.

<table>
<thead>
<tr>
<th>Table 25-6 Signature Digest Tag Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>-------------------------</td>
</tr>
<tr>
<td>RPMSIGTAG_SHA1</td>
</tr>
<tr>
<td>RPMSIGTAG_MD5</td>
</tr>
</tbody>
</table>

RPMSIGTAG_SHA1
This index contains the SHA1 checksum of the entire Header Section, including the Header Record, Index Records and Header store.

RPMSIGTAG_MD5
This tag specifies the 128-bit MD5 checksum of the combined Header and Archive
sections.

These values exist to provide authentication of the package.

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMSIGTAG_DSA</td>
<td>267</td>
<td>BIN</td>
<td>65</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMSIGTAG_RSA</td>
<td>268</td>
<td>BIN</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMSIGTAG_PGP</td>
<td>1002</td>
<td>BIN</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMSIGTAG_GPG</td>
<td>1005</td>
<td>BIN</td>
<td>65</td>
<td>Optional</td>
</tr>
</tbody>
</table>

RPMSIGTAG_DSA

The tag contains the DSA signature of the Header section. The data is formatted as a Version 3 Signature Packet as specified in [RFC 2440: OpenPGP Message Format](https://tools.ietf.org/html/rfc2440). If this tag is present, then the SIGTAG_GPG tag shall also be present.

RPMSIGTAG_RSA

The tag contains the RSA signature of the Header section. The data is formatted as a Version 3 Signature Packet as specified in [RFC 2440: OpenPGP Message Format](https://tools.ietf.org/html/rfc2440). If this tag is present, then the SIGTAG_PGP shall also be present.

RPMSIGTAG_PGP

This tag specifies the RSA signature of the combined Header and Payload sections. The data is formatted as a Version 3 Signature Packet as specified in [RFC 2440: OpenPGP Message Format](https://tools.ietf.org/html/rfc2440).

RPMSIGTAG_GPG

The tag contains the DSA signature of the combined Header and Payload sections. The data is formatted as a Version 3 Signature Packet as specified in [RFC 2440: OpenPGP Message Format](https://tools.ietf.org/html/rfc2440).

25.2.4 Header Section

The Header section is implemented using the Header structure. The Header section defines the following additional tag values which may be used in the Index structures.

25.2.4.1 Package Information

The following tag values are used to indicate information that describes the package as a whole.

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_NAME</td>
<td>1000</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_VERSION</td>
<td>1001</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_RELEASE</td>
<td>1002</td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_SUMMARY</td>
<td>1004</td>
<td>I18NSTRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_DESC</td>
<td>1005</td>
<td>I18NSTRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>Tag</td>
<td>Description</td>
<td>Format</td>
<td>Size</td>
<td>Required/Informational</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------</td>
<td>---------</td>
<td>------</td>
<td>------------------------</td>
</tr>
<tr>
<td>RPMTAG_SIZED</td>
<td></td>
<td>INT32</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_DISTRIBUTION</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Informational</td>
</tr>
<tr>
<td>RPMTAG_VENDOR</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Informational</td>
</tr>
<tr>
<td>RPMTAG_LICENSE</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_PACKAGE</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Informational</td>
</tr>
<tr>
<td>RPMTAG_GROUP</td>
<td></td>
<td>I18NSTRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_URL</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Informational</td>
</tr>
<tr>
<td>RPMTAG_OS</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_ARCH</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_SOURCERPM</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Informational</td>
</tr>
<tr>
<td>RPMTAG_ARCHIVESIZE</td>
<td></td>
<td>INT32</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_RPMVERSION</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Informational</td>
</tr>
<tr>
<td>RPMTAG_COOKIE</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_DISTRIBUTURL</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Informational</td>
</tr>
<tr>
<td>RPMTAG_PAYLOADFORMAT</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_PAYLOADCOMPRESSOR</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_PAYLOADFLAGS</td>
<td></td>
<td>STRING</td>
<td>1</td>
<td>Required</td>
</tr>
</tbody>
</table>

**RPMTAG_NAME**

This tag specifies the name of the package.

**RPMTAG_VERSION**

This tag specifies the version of the package.

**RPMTAG_RELEASE**

This tag specifies the release of the package.

**RPMTAG_SUMMARY**

This tag specifies the summary description of the package. The summary value pointed to by this index record contains a one line description of the package.

**RPMTAG_DESCRIPTION**

This tag specifies the description of the package. The description value pointed to by this index record contains a full description of the package.
RPMTAG_SIZE
This tag specifies the sum of the sizes of the regular files in the archive.

RPMTAG_DISTRIBUTION
A string containing the name of the distribution on which the package was built.

RPMTAG_VENDOR
A string containing the name of the organization that produced the package.

RPMTAG_LICENSE
This tag specifies the license which applies to this package.

RPMTAG_PACKAGER
A string identifying the tool used to build the package.

RPMTAG_GROUP
This tag specifies the administrative group to which this package belongs.

RPMTAG_URL
Generic package information URL.

RPMTAG_OS
This tag specifies the OS of the package. The OS value pointed to by this index record shall be “linux”.

RPMTAG_ARCH
This tag specifies the architecture of the package. The architecture value pointed to by this index record is defined in architecture specific LSB specification.

RPMTAG_SOURCERPM
This tag specifies the name of the source RPM.

RPMTAG_ARCHIVESIZE
This tag specifies the uncompressed size of the Payload archive, including the cpio headers.

RPMTAG_RPMVERSION
This tag indicates the version of RPM tool used to build this package. The value is unused.

RPMTAG_COOKIE
This tag contains an opaque string whose contents are undefined.

RPMTAG_DISTURL
URL for package.

RPMTAG_PAYLOADFORMAT
This tag specifies the format of the Archive section. The format value pointed to by this index record shall be ‘cpio’.

RPMTAG_PAYLOADCOMPRESSOR
This tag specifies the compression used on the Archive section. The compression value pointed to by this index record shall be ‘gzip’.
RPMTAG_PAYLOADFLAGS

This tag indicates the compression level used for the Payload. This value shall always be '9'.

25.2.4.2 Installation Information

The following tag values are used to provide information needed during the installation of the package.

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_PREIN</td>
<td>1023</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_POSTIN</td>
<td>1024</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_PREUN</td>
<td>1025</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_POSTUN</td>
<td>1026</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_PREINPROG</td>
<td>1085</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_POSTINPROG</td>
<td>1086</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_PREUNPROG</td>
<td>1087</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_POSTUNPROG</td>
<td>1088</td>
<td>STRING</td>
<td>1</td>
<td>Optional</td>
</tr>
</tbody>
</table>

RPMTAG_PREIN

This tag specifies the preinstall scriptlet. If present, then RPMTAG_PREINPROG shall also be present.

RPMTAG_POSTIN

This tag specifies the postinstall scriptlet. If present, then RPMTAG_POSTINPROG shall also be present.

RPMTAG_PREUN

This tag specifies the preuninstall scriptlet. If present, then RPMTAG_PREUNPROG shall also be present.

RPMTAG_POSTUN

This tag specifies the postuninstall scriptlet. If present, then RPMTAG_POSTUNPROG shall also be present.

RPMTAG_PREINPROG

This tag specifies the name of the interpreter to which the preinstall scriptlet will be passed. The interpreter pointed to by this index record shall be /bin/sh.

RPMTAG_POSTINPROG

This tag specifies the name of the interpreter to which the postinstall scriptlet will be passed. The interpreter pointed to by this index record shall be /bin/sh.

RPMTAG_PREUNPROG

This tag specifies the name of the interpreter to which the preuninstall scriptlet will be passed. The interpreter pointed to by this index record shall be /bin/sh.
be passed. The interpreter pointed to by this index record shall be /bin/sh.

RPMTAG_POSTUNPROG

This program specifies the name of the interpreter to which the postuninstall scriptlet will be passed. The interpreter pointed to by this index record shall be /bin/sh.

25.2.4.3 File Information

The following tag values are used to provide information about the files in the payload. This information is provided in the header to allow more efficient access of the information.

**Table 25-10 File Info Tag Values**

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_OLD_FILENAMES</td>
<td>1027</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_FILEESIZES</td>
<td>1028</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILEEMODES</td>
<td>1030</td>
<td>INT16</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILEERDEVS</td>
<td>1033</td>
<td>INT16</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_FILEETIME</td>
<td>1034</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG/FileEMDSS</td>
<td>1035</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG/FileELINKTOS</td>
<td>1036</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG/FileEFLAGS</td>
<td>1037</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG/FileEUSERNAME</td>
<td>1039</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG/FileEGROUPNAME</td>
<td>1040</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG/FileEDEVICES</td>
<td>1095</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG/FileEINODES</td>
<td>1096</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG/FileELANGS</td>
<td>1097</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_DIRINDEXES</td>
<td>1116</td>
<td>INT32</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_DIRENAMES</td>
<td>1117</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_DIRNAMES</td>
<td>1118</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
</tbody>
</table>

RPMTAG_OLDFILENAMES

This tag specifies the filenames when not in a compressed format as determined by the absence of rpmlib(CompressedFileNames) in the RPMTAG_REQUIRENAME index.
RPMTAG_FILESIZES
This tag specifies the size of each file in the archive.

RPMTAG_FILEMODES
This tag specifies the mode of each file in the archive.

RPMTAG_FILERDEVS
This tag specifies the device number from which the file was copied.

RPMTAG_FILETIMES
This tag specifies the modification time in seconds since the epoch of each file in the archive.

RPMTAG_FILEMD5S
This tag specifies the ASCII representation of the MD5 sum of the corresponding file contents. This value is empty if the corresponding archive entry is not a regular file.

RPMTAG_FILELINKTOS
The target for a symlink, otherwise NULL.

RPMTAG_FILEFLAGS
This tag specifies the bit(s) to classify and control how files are to be installed. See below.

RPMTAG_FILEUSERNAME
This tag specifies the owner of the corresponding file.

RPMTAG_FILEGROUPNAME
This tag specifies the group of the corresponding file.

RPMTAG_FILEDEVICES
This tag specifies the 16 bit device number from which the file was copied.

RPMTAG_FILEINODES
This tag specifies the inode value from the original file system on the system on which it was built.

RPMTAG_FILELANGS
This tag specifies a per-file locale marker used to install only locale specific subsets of files when the package is installed.

RPMTAG_DIRINDEXES
This tag specifies the index into the array provided by the RPMTAG_DIRNAMES Index which contains the directory name for the corresponding filename.

RPMTAG_BASENAMES
This tag specifies the base portion of the corresponding filename.

RPMTAG_DIRNAMES

One of RPMTAG_OLDFILENAMES or the tuple RPMTAG_DIRINDEXES, RPMTAG_BASENAMES, RPMTAG_DIRNAMES shall be present, but not both.
25.2.4.3.1 File Flags

The RPMTAG_FILEFLAGS tag value shall identify various characteristics of the file in the payload that it describes. It shall be an INT32 value consisting of either the value RPMFILE_NONE (0) or the bitwise inclusive or of one or more of the following values:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMFILE_CONFIG</td>
<td>(1 &lt;&lt; 0)</td>
</tr>
<tr>
<td>RPMFILE_DOC</td>
<td>(1 &lt;&lt; 1)</td>
</tr>
<tr>
<td>RPMFILE_DONOTUSE</td>
<td>(1 &lt;&lt; 2)</td>
</tr>
<tr>
<td>RPMFILE_MISSINGOK</td>
<td>(1 &lt;&lt; 3)</td>
</tr>
<tr>
<td>RPMFILE_NOREPLACE</td>
<td>(1 &lt;&lt; 4)</td>
</tr>
<tr>
<td>RPMFILE_SPECFILE</td>
<td>(1 &lt;&lt; 5)</td>
</tr>
<tr>
<td>RPMFILE_GHOST</td>
<td>(1 &lt;&lt; 6)</td>
</tr>
<tr>
<td>RPMFILE_LICENSE</td>
<td>(1 &lt;&lt; 7)</td>
</tr>
<tr>
<td>RPMFILE_README</td>
<td>(1 &lt;&lt; 8)</td>
</tr>
<tr>
<td>RPMFILE_EXCLUDE</td>
<td>(1 &lt;&lt; 9)</td>
</tr>
</tbody>
</table>

These bits have the following meaning:

RPMFILE_CONFIG
The file is a configuration file, and an existing file should be saved during a package upgrade operation and not removed during a package removal operation.

RPMFILE_DOC
The file contains documentation.

RPMFILE_DONOTUSE
This value is reserved for future use; conforming packages may not use this flag.

RPMFILE_MISSINGOK
The file need not exist on the installed system.

RPMFILE_NOREPLACE
Similar to the RPMFILE_CONFIG, this flag indicates that during an upgrade operation the original file on the system should not be altered.

RPMFILE_SPECFILE
The file is a package specification.

RPMFILE_GHOST
The file is not actually included in the payload, but should still be considered as a part of the package. For example, a log file generated by the application at run time.

RPMFILE_LICENSE
The file contains the license conditions.

RPMFILE_README
The file contains high level notes about the package.
RPMFILE_EXCLUDE

The corresponding file is not a part of the package, and should not be installed.

25.2.4.4 Dependency Information

The following tag values are used to provide information about interdependencies between packages.

Table 25-12 Package Dependency Tag Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_PROVIDENAME</td>
<td>1047</td>
<td>STRING_ARRAY</td>
<td>1</td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_REQUIREFLAGS</td>
<td>1048</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_REQUIRENAME</td>
<td>1049</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_REQUIREVERSION</td>
<td>1050</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_CONFLICTFLAGS</td>
<td>1053</td>
<td>INT32</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_CONFLICTNAME</td>
<td>1054</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_CONFLICTVERSION</td>
<td>1055</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_OBSOLETENAME</td>
<td>1090</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_PROVIDENAME</td>
<td>1112</td>
<td>INT32</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_PROVIDEVERSION</td>
<td>1113</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Required</td>
</tr>
<tr>
<td>RPMTAG_OBSOLETEFLAGS</td>
<td>1114</td>
<td>INT32</td>
<td>1</td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_OBSOLETEVERSION</td>
<td>1115</td>
<td>STRING_ARRAY</td>
<td></td>
<td>Optional</td>
</tr>
</tbody>
</table>

RPMTAG_PROVIDENAME

This tag indicates the name of the dependency provided by this package.

RPMTAG_REQUIREFLAGS

Bits(s) to specify the dependency range and context.

RPMTAG_REQUIRENAME

This tag indicates the dependencies for this package.

RPMTAG_REQUIREVERSION

This tag indicates the versions associated with the values found in the RPMTAG_REQUIRENAME Index.

RPMTAG_CONFLICTFLAGS

Bits(s) to specify the conflict range and context.
25 Software Installation

**LSB Core - Generic 5.0**

RPMTAG_CONFLICTNAME
This tag indicates the conflicting dependencies for this package.

RPMTAG_CONFLICTVERSION
This tag indicates the versions associated with the values found in the RPMTAG_CONFLICTNAME Index.

RPMTAG_OBSOLETENAME
This tag indicates the obsoleted dependencies for this package.

RPMTAG_PROVIDEFLAGS
Bits(s) to specify the conflict range and context.

RPMTAG_PROVIDEVERSION
This tag indicates the versions associated with the values found in the RPMTAG_PROVIDENAME Index.

RPMTAG_OBSOLETEFLAGS
Bits(s) to specify the conflict range and context.

RPMTAG_OBSOLETEVERSION
This tag indicates the versions associated with the values found in the RPMTAG_OBSOLETENAME Index.

### 25.2.4.4.1 Package Dependency Values

The package dependencies are stored in the RPMTAG_REQUIRENAME and RPMTAG_REQUIREVERSION index records. The following values may be used.

**Table 25-13 Index Type values**

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
<th>Meaning</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>rpmlib(Versioned Dependencies)</td>
<td>3.0.3-1</td>
<td>Indicates that the package contains RPMTAG_PROVIDENAME, RPMTAG_OBSOLETENAME or RPMTAG_PREREQ records that have a version associated with them.</td>
<td>Optional</td>
</tr>
<tr>
<td>rpmlib(PayloadFilesHavePrefix)</td>
<td>4.0-1</td>
<td>Indicates the filenames in the Archive have had “.” prepended to them.</td>
<td>Optional</td>
</tr>
<tr>
<td>rpmlib(CompressedFileNames)</td>
<td>3.0.4-1</td>
<td>Indicates that the filenames in the Payload are represented in the RPMTAG_DIRINDEXES, RPMTAG_DIRNAME and RPMTAG_BASENAMES indexes.</td>
<td>Optional</td>
</tr>
</tbody>
</table>
Additional dependencies are specified in the Package Dependencies section of this specification, and in the relevant architecture specific part of the LSB Core Specification.

25.2.4.4.2 Package Dependencies Attributes

The package dependency attributes are stored in the RPMTAG_REQUIREFLAGS, RPMTAG_PROVIDEFLAGS and RPMTAG_OBSOLETEFLAGS index records. The following values may be used.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMSENSE_LESS</td>
<td>0x02</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_GREATER</td>
<td>0x04</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_EQUAL</td>
<td>0x08</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_PREREQ</td>
<td>0x40</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_INTERP</td>
<td>0x100</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_SCRIPT_PRE</td>
<td>0x200</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_SCRIPT_POS</td>
<td>0x400</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_SCRIPT_PRE_UN</td>
<td>0x800</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_SCRIPT_POS_UN</td>
<td>0x1000</td>
<td></td>
</tr>
<tr>
<td>RPMSENSE_RPMLIB</td>
<td>0x1000000</td>
<td></td>
</tr>
</tbody>
</table>

25.2.4.5 Other Information

The following tag values are also found in the Header section.

<table>
<thead>
<tr>
<th>Name</th>
<th>Tag Value</th>
<th>Type</th>
<th>Count</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPMTAG_BUILDLTIME</td>
<td>1006</td>
<td>INT32</td>
<td>1</td>
<td>Informational</td>
</tr>
<tr>
<td>RPMTAG_BUILDLHOST</td>
<td>1007</td>
<td>STRING</td>
<td>1</td>
<td>Informational</td>
</tr>
<tr>
<td>RPMTAG_FILEVERIFYFLAGS</td>
<td>1045</td>
<td>INT32</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_CHANGELOGTIME</td>
<td>1080</td>
<td>INT32</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_CHANGELOGNAME</td>
<td>1081</td>
<td>STRING_ARR</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_CHANGELOGTEXT</td>
<td>1082</td>
<td>STRING_ARR</td>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>RPMTAG_OPTFLAGS</td>
<td>1122</td>
<td>STRING</td>
<td>1</td>
<td>Informational</td>
</tr>
<tr>
<td>RPMTAG_RHNPLATFORM</td>
<td>1131</td>
<td>STRING</td>
<td>1</td>
<td>Deprecated</td>
</tr>
<tr>
<td>RPMTAG_PLATFORM</td>
<td>1132</td>
<td>STRING</td>
<td>1</td>
<td>Informational</td>
</tr>
</tbody>
</table>
RPMTAG_BUILDTIME

This tag specifies the time as seconds since the epoch at which the package was built.

RPMTAG_BUILDHOST

This tag specifies the hostname of the system on which the package was built.

RPMTAG_FILEVERIFYFLAGS

This tag specifies the bit(s) to control how files are to be verified after install, specifying which checks should be performed.

RPMTAG_CHANGELOGTIME

This tag specifies the Unix time in seconds since the epoch associated with each entry in the Changelog file.

RPMTAG_CHANGELOGNAME

This tag specifies the name of who made a change to this package.

RPMTAG_CHANGELOGTEXT

This tag specifies the changes associated with a changelog entry.

RPMTAG_OPTFLAGS

This tag indicates additional flags which may have been passed to the compiler when building this package.

RPMTAG_RHNPLATFORM

This tag contains an opaque string whose contents are undefined.

RPMTAG_PLATFORM

This tag contains an opaque string whose contents are undefined.

25.2.5 Payload Section

The Payload section contains a compressed cpio archive. The format of this section is defined by RFC 1952: GZIP File Format Specification.

When uncompressed, the cpio archive contains a sequence of records for each file. Each record contains a CPIO Header, Filename, Padding, and File Data.

<table>
<thead>
<tr>
<th>Table 25-16 CPIO File Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPIO Header</td>
</tr>
<tr>
<td>Filename</td>
</tr>
<tr>
<td>Padding</td>
</tr>
<tr>
<td>File data</td>
</tr>
<tr>
<td>Padding</td>
</tr>
</tbody>
</table>

The CPIO Header uses the following header structure (sometimes referred to as "new ASCII" or "SVR4 cpio"). All numbers are stored as ASCII representations of their hexadecimal value with leading zeros as needed to fill the field. With the exception of c_namesize and the corresponding name string, and c_checksum, all information contained in the CPIO Header is also represented in the Header Section. The values in
the CPIO Header shall match the values contained in the Header Section.

```c
struct {
    char c_magic[6];
    char c_ino[8];
    char c_mode[8];
    char c_uid[8];
    char c_gid[8];
    char c_nlink[8];
    char c_mtime[8];
    char c_filesize[8];
    char c_devmajor[8];
    char c_devminor[8];
    char c_rdevmajor[8];
    char c_rdevminor[8];
    char c_namesize[8];
    char c_checksum[8];
};
```

**c_magic**

Value identifying this cpio format. This value shall be "070701".

**c_ino**

This field contains the inode number from the filesystem from which the file was read. This field is ignored when installing a package. This field shall match the corresponding value in the `RPMTAG_FILEINODES` index in the Header section.

**c_mode**

Permission bits of the file. This is an ascii representation of the hexadecimal number representing the bit as defined for the `st_mode` field of the `stat` structure defined for the `stat` function. This field shall match the corresponding value in the `RPMTAG_FILEMODES` index in the Header section.

**c_uid**

Value identifying this owner of this file. This value matches the uid value of the corresponding user in the `RPMTAG_FILEUSERNAME` as found on the system where this package was built. The username specified in `RPMTAG_FILEUSERNAME` should take precedence when installing the package.

**c_gid**

Value identifying this group of this file. This value matches the gid value of the corresponding user in the `RPMTAG_FILEGROUPNAME` as found on the system where this package was built. The groupname specified in `RPMTAG_FILEGROUPNAME` should take precedence when installing the package.

**c_nlink**

Value identifying the number of links associated with this file. If the value is greater than 1, then this filename will be linked to 1 or more files in this archive that has a matching value for the `c_ino`, `c_devmajor` and `c_devminor` fields.

**c_mtime**

Value identifying the modification time of the file when it was read. This field shall match the corresponding value in the `RPMTAG_FILETIMES` index in the Header section.

**c_filesize**

Value identifying the size of the file. This field shall match the corresponding value
in the RPMTAG_FILESIZES index in the Header section.

\texttt{c\_devmajor}

The major number of the device containing the file system from which the file was read. With the exception of processing files with \texttt{c\_nlink > 1}, this field is ignored when installing a package. This field shall match the corresponding value in the RPMTAG_FILEDEVICES index in the Header section.

\texttt{c\_devminor}

The minor number of the device containing the file system from which the file was read. With the exception of processing files with \texttt{c\_nlink > 1}, this field is ignored when installing a package. This field shall match the corresponding value in the RPMTAG_FILEDEVICES index in the Header section.

\texttt{c\_rdevmajor}

The major number of the raw device containing the file system from which the file was read. This field is ignored when installing a package. This field shall match the corresponding value in the RPMTAG_RDEVS index in the Header section.

\texttt{c\_rdevminor}

The minor number of the raw device containing the file system from which the file was read. This field is ignored when installing a package. This field shall match the corresponding value in the RPMTAG_RDEVS index in the Header section.

\texttt{c\_namesize}

Value identifying the length of the filename, which is located immediately following the CPIO Header structure.

\texttt{c\_checksum}

Value containing the CRC checksum of the file data. This field is not used, and shall contain the value "00000000". This field is ignored when installing a package.

A record with the filename "TRAILER!!" indicates the last record in the archive.

25.3 Package Script Restrictions

Scripts used as part of the package install and uninstall shall only use commands and interfaces that are specified by the LSB. All other commands are not guaranteed to be present, or to behave in expected ways.

Packages shall not use RPM triggers.

Packages shall not depend on the order in which scripts are executed (pre-install, pre-uninstall, etc), when doing an upgrade.

25.4 Package Tools

The LSB does not specify the interface to the tools used to manipulate LSB-conformant packages. Each conforming implementation shall provide documentation for installing LSB packages.

25.5 Package Naming Conventions

Packages supplied by distributions and applications should adhere to the following conventions for the name field within the package. The rules are optional for the filename of the package file itself.

\textbf{Note}: There are discrepancies among implementations concerning whether the name might be \texttt{frobnicator-1.7-21-ppc32.rpm} or \texttt{frobnicator-1.7-21-powerpc32.rpm}. The architecture aside, recommended practice is for the filename of the package file to match the
name within the package.

The following conventions apply to the name portion of the field alone, not including any release or version portion.

**Note:** If the package name with the release and version is `frobnicator-1.7-21`, the name part is `frobnicator` and falls under the conventions for a name with no hyphens.

- If the name begins with `lsb-` and contains no other hyphens, the name should be a package name registered with the Linux Assigned Names and Numbers Authority (http://www.lanana.org) (LANANA), which shall maintain a registry of LSB names. The name may be registered by either an implementation or an application.

- If the name begins with `lsb-` and contains more than one hyphen the portion of the name between the first and second hyphens should be either an LSB provider name registered with the LANANA (for example `lsb-gnome-gnumeric` if `gnome` is registered), or a domain name registered to the provider in the DNS system. (for example `lsb-distro.example.com-database`). The LSB provider name registered with the LANANA shall only consist of the ASCII characters `[a-z0-9]`. The domain name, in accordance with DNS rules, shall be lower case only. The provider name or domain name may be either that of a distribution or an application.

- Package names containing no hyphens are reserved for use by distributions. Applications shall not use such names.

- Package names which do not start with `lsb-` and which contain a hyphen are open to both distributions and applications. Distributions may name packages in any part of this namespace. They are encouraged to use names from one of the other namespaces available to them, but this is not mandatory due to the large amount of current practice to the contrary.

  **Note:** Widespread existing practice includes such names as `ssh-common`, `ssh-client`, `kernel-pcmcia`, and the like. Possible alternative names include `sshc`, `foolinux-ssh-common` (where `foolinux` is registered to the distribution), or `lsb-foolinux-ssh-common`.

Applications may name their packages this way, but only if the portion of the name before the first hyphen is a provider name or registered domain name as described above.

  **Note:** If an application vendor has domain name such as `visicalc.example.com` and has registered `visicalc` as a provider name, they could name packages either in the `visicalc-base` style or the `visicalc.example.com-charting` style.

Package names in this namespace are available to both the distribution and an application. Distributions and applications need to consider the potential for conflicts when deciding to use these names rather than the alternatives (such as names starting with `lsb-`).

### 25.6 Package Dependencies

Packages shall have a dependency that indicates which LSB modules are required. LSB module descriptions are dash separated tuples containing the name 'lsb', the module name, and the architecture name. The following dependencies may be used.

**lsb-core**

This dependency is used to indicate that the application is dependent on features contained in the LSB Core specification.

**lsb-core-arch**

This dependency is used to indicate that the application is dependent on features contained in the LSB Core specification and that the package contains architecture specific features. This architecture specific dependency is described in the relevant
architecture specific part of the LSB Core specification.

lsb-core-noarch

This dependency is used to indicate that the application is dependent on features contained in the LSB Core specification and that the package does not contain any architecture specific files.

These dependencies shall have a version of 5.0.

Packages shall not depend on other system-provided dependencies. They shall not depend on non-system-provided dependencies unless the package provider also makes available the LSB conforming packages needed to satisfy such dependencies.

Other modules in the LSB may supplement this list. The architecture specific dependencies are described in the relevant architecture specific LSB.

25.7 Package Architecture Considerations

Packages which do not contain any architecture specific files should specify an architecture of noarch. An LSB runtime environment shall accept values noarch, or the value specified in the relevant architecture specific part of the LSB Core Specification.

Additional specifications or restrictions may be found in the architecture specific LSB specification.
LSB Core - Generic 5.0

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]
This Specification [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[SUSv4]

getdelim[SUSv4]

scandir[SUSv4]

_IO_feof[LSB]

getdomainname[LSB]

scandir64[LSB]

_IO_getc[LSB]

getdtablesize[LSB]

scanf[LSB]

_IO_putc[LSB]

getegid[SUSv4]

sched_get_priority_max[
SUSv4]

_IO_puts[LSB]

getenv[SUSv4]

sched_get_priority_min[S
USv4]

__assert_fail[LSB]

geteuid[SUSv4]

sched_getaffinity(GLIBC
_2.3.4)[LSB]

__chk_fail(GLIBC_2.3.4)
[LSB]

getgid[SUSv4]

sched_getparam[SUSv4]

__confstr_chk(GLIBC_2.
4)[LSB]

getgrent[SUSv4]

sched_getscheduler[SUSv
4]

__ctype_b_loc(GLIBC_2.
3)[LSB]

getgrent_r[LSB]

sched_rr_get_interval[SU
Sv4]

__ctype_get_mb_cur_ma
x[LSB]

getgrgid[SUSv4]

sched_setaffinity(GLIBC
_2.3.4)[LSB]

__ctype_tolower_loc(GLI
BC_2.3)[LSB]

getgrgid_r[SUSv4]

sched_setparam[SUSv4]

__ctype_toupper_loc(GLI
BC_2.3)[LSB]

getgrnam[SUSv4]

sched_setscheduler[LSB]

__cxa_atexit[LSB]

getgrnam_r[SUSv4]

sched_yield[SUSv4]

__cxa_finalize[LSB]

getgrouplist[LSB]

seed48[SUSv4]

__errno_location[LSB]

getgroups[SUSv4]

seed48_r[LSB]

__fgets_chk(GLIBC_2.4)
[LSB]

gethostbyaddr[SUSv3]

seekdir[SUSv4]

__fgets_unlocked_chk(G
LIBC_2.4)[LSB]

gethostbyaddr_r[LSB]

select[SUSv4]

__fgetws_chk(GLIBC_2.
4)[LSB]

gethostbyname[SUSv3]

semctl[SUSv4]

__fgetws_unlocked_chk(
GLIBC_2.4)[LSB]

gethostbyname2[LSB]

semget[SUSv4]

__fpending[LSB]

gethostbyname2_r[LSB]

semop[SUSv4]

__fprintf_chk[LSB]

gethostbyname_r[LSB]

send[SUSv4]

__fwprintf_chk(GLIBC_2
.4)[LSB]

gethostid[SUSv4]

sendfile[LSB]

© 2015 Linux Foundation

931


<table>
<thead>
<tr>
<th>Function</th>
<th>Library/Version</th>
<th>Function</th>
<th>Library/Version</th>
<th>Function</th>
<th>Library/Version</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>__fxstat[LSB]</code></td>
<td></td>
<td><code>gethostname[SUSv4]</code></td>
<td></td>
<td><code>sendfile64(GLIBC_2.3)</code></td>
<td><code>[LSB]</code></td>
</tr>
<tr>
<td><code>__fxstat64[LSB]</code></td>
<td></td>
<td><code>getifaddrs(GLIBC_2.3)</code></td>
<td><code>[LSB]</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__fxstatat(GLIBC_2.4)[LSB]</code></td>
<td></td>
<td><code>gethostname[SUSv4]</code></td>
<td></td>
<td><code>sendmsg[SUSv4]</code></td>
<td></td>
</tr>
<tr>
<td><code>__fxstat64at(GLIBC_2.4)[LSB]</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__getcwd_chk(GLIBC_2.4)</code></td>
<td></td>
<td><code>getitimer[SUSv4]</code></td>
<td></td>
<td><code>sendto[SUSv4]</code></td>
<td></td>
</tr>
<tr>
<td><code>__getgroups_chk(GLIBC_2.4)</code></td>
<td></td>
<td><code>getline[SUSv4]</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__gethostname_chk(GLIBC_2.4)</code></td>
<td></td>
<td><code>getloadavg[LSB]</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__getlogin_r_chk(GLIBC_2.4)</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__getpagesize</code></td>
<td></td>
<td><code>getopt[LSB]</code></td>
<td></td>
<td><code>setcontext[SUSv3]</code></td>
<td></td>
</tr>
<tr>
<td><code>__getpgid</code></td>
<td></td>
<td><code>getopt_long[LSB]</code></td>
<td></td>
<td><code>seteuid[SUSv4]</code></td>
<td></td>
</tr>
<tr>
<td><code>__h_errno_location</code></td>
<td></td>
<td><code>getopt_long_only[LSB]</code></td>
<td></td>
<td><code>setgid[SUSv4]</code></td>
<td></td>
</tr>
<tr>
<td><code>__isinf</code></td>
<td></td>
<td><code>getpagesize[LSB]</code></td>
<td></td>
<td><code>setgrent[SUSv4]</code></td>
<td></td>
</tr>
<tr>
<td><code>__isinff</code></td>
<td></td>
<td><code>getpeername[LSB]</code></td>
<td></td>
<td><code>setgroups[LSB]</code></td>
<td></td>
</tr>
<tr>
<td><code>__isnan</code></td>
<td></td>
<td><code>getpgid[LSB]</code></td>
<td></td>
<td><code>sethostname[LSB]</code></td>
<td></td>
</tr>
<tr>
<td><code>__isnanf</code></td>
<td></td>
<td><code>getpwent</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__isnanl</code></td>
<td></td>
<td><code>getpwuid</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__libc_current_sigrtmax[LSB]</code></td>
<td></td>
<td><code>getpriority[SUSv4]</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__libc_current_sigrtmin[LSB]</code></td>
<td></td>
<td><code>getprotobynumber[SUSv4]</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__libc_start_main[LSB]</code></td>
<td></td>
<td><code>getprotobynumber_r[LSB]</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__lxstat</code></td>
<td></td>
<td><code>getprotobyname[SUSv4]</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__lxstat64</code></td>
<td></td>
<td><code>getprotobyname_r[LSB]</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__mbsnrtowcs_chk(GLIBC_2.4)[LSB]</code></td>
<td></td>
<td><code>getprotoent[SUSv4]</code></td>
<td></td>
<td><code>setregid[SUSv4]</code></td>
<td></td>
</tr>
<tr>
<td><code>__mbsrtowcs_chk(GLIBC_2.4)[LSB]</code></td>
<td></td>
<td><code>getprotoent_r[LSB]</code></td>
<td></td>
<td><code>setreuid[SUSv4]</code></td>
<td></td>
</tr>
<tr>
<td><code>__mbstowcs_chk(GLIBC_2.4)[LSB]</code></td>
<td></td>
<td><code>getpwent</code></td>
<td></td>
<td><code>setrlimit64[LFS]</code></td>
<td></td>
</tr>
<tr>
<td><code>__memcpy_chk(GLIBC_2.3.4)</code></td>
<td></td>
<td><code>getpwuid</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__memcpy_chk(GLIBC_2.4)</code></td>
<td></td>
<td><code>getpwuid_r[LSB]</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__mempcpy</code></td>
<td></td>
<td><code>getpwuid[LSB]</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__mempcpy chk(GLIBC_2.3.4)[LSB]</code></td>
<td></td>
<td><code>getpwnam</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>__mempcpy chk(GLIBC_2.4)[LSB]</code></td>
<td></td>
<td><code>getpwnam_r[LSB]</code></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Library</td>
<td>Library</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------------</td>
<td>------------------</td>
<td>------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__pread64_chk(GLIBC_2.4)</td>
<td>GLIBC_2.4</td>
<td>GLIBC_2.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__pread_chk(GLIBC_2.4)</td>
<td>GLIBC_2.4</td>
<td>GLIBC_2.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>_preff_chk(GLIBC_2.4)</td>
<td>GLIBC_2.4</td>
<td>GLIBC_2.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__readlink chk(GLIBC_2.4)</td>
<td>GLIBC_2.4</td>
<td>GLIBC_2.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__realpath chk(GLIBC_2.4)</td>
<td>GLIBC_2.4</td>
<td>GLIBC_2.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__realpath_chk(GLIBC_2.4)</td>
<td>GLIBC_2.4</td>
<td>GLIBC_2.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__readfrom chk(GLIBC_2.4)</td>
<td>GLIBC_2.4</td>
<td>GLIBC_2.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__register_atfork(GLIBC_2.3.2)</td>
<td>GLIBC_2.3.2</td>
<td>GLIBC_2.3.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__sigsetjmp</td>
<td>GLIBC_2.4</td>
<td>GLIBC_2.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__snprintf_chk</td>
<td>GLIBC_2.4</td>
<td>GLIBC_2.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__stpcpy</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__stpncpy</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strcat</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strcpy</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strdup</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strncat</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strncpy</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strndup</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strtol</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strtoll</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strtoul</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__swprintf_chk</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__stdiof_internal</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strtok</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strtolf</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strtofs</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strtooll</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__strtoulf</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__swprintf_chk</td>
<td>GLIBC_2.3.4</td>
<td>GLIBC_2.3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Library</td>
<td>Interface</td>
<td>Function</td>
<td>Library</td>
<td>Function</td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------</td>
<td>------------</td>
<td>-----------------</td>
<td>------------</td>
<td></td>
</tr>
<tr>
<td>LSB Core - Generic 5.0</td>
<td><strong>sysconf</strong></td>
<td>gmtime[SUSv4]</td>
<td>sigsuspend[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__syslog_chk(GLIBC_2.4)</td>
<td>gmtime_r[SUSv4]</td>
<td>sigtimedwait[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__sysv_signal</td>
<td>gnu_get_libc_release[LSB]</td>
<td>sigwait[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__ttyname_r_chk(GLIBC_2.4)</td>
<td>gnu_get_libc_version[LSB]</td>
<td>sigwaitinfo[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__vfprintf_chk</td>
<td>grantpt[SUSv4]</td>
<td>sleep[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__vfwprintf_chk(GLIBC_2.4)</td>
<td>hcreate[SUSv4]</td>
<td>snprintf[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__vprintf_chk</td>
<td>hcreate_r[LSB]</td>
<td>sockatmark[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__vsprintf_chk</td>
<td>hdestroy[LSB]</td>
<td>socket[LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__vsnprintf_chk</td>
<td>hdestroy_r[LSB]</td>
<td>socketpair[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__vsyslog_chk(GLIBC_2.4)</td>
<td>hsearch[LSB]</td>
<td>sprintf[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__vwprintf_chk(GLIBC_2.4)</td>
<td>hsearch_r[LSB]</td>
<td>srand[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcpcpy_chk(GLIBC_2.4)</td>
<td>htonl[SUSv4]</td>
<td>srand48[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcpncpy_chk(GLIBC_2.4)</td>
<td>htons[SUSv4]</td>
<td>srand48_r[LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcrtomb_chk(GLIBC_2.4)</td>
<td>icnv[SUSv4]</td>
<td>srandom[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcscat_chk(GLIBC_2.4)</td>
<td>icnv_close[SUSv4]</td>
<td>srandom_r[LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcscpy_chk(GLIBC_2.4)</td>
<td>icnv_open[SUSv4]</td>
<td>sscanf[LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcsncat_chk(GLIBC_2.4)</td>
<td>if_freenameindex[SUSv4]</td>
<td>statfs[LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcsncpy_chk(GLIBC_2.4)</td>
<td>if_indextoname[SUSv4]</td>
<td>statfs64[LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcsnrtombs_chk(GLIBC_2.4)</td>
<td>if_nameindex[SUSv4]</td>
<td>statvfs64[LFS]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcsrtombs_chk(GLIBC_2.4)</td>
<td>imaxabs[SUSv4]</td>
<td>stime[LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcstod_internal</td>
<td>imaxdiv[SUSv4]</td>
<td>stpcpy[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcstrof_internal</td>
<td>index[SUSv3]</td>
<td>stpncpy[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcstol_internal</td>
<td>inet_addr[SUSv4]</td>
<td>strcasecmp[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcstold_internal</td>
<td>inet_aton[LSB]</td>
<td>strcasecmp_l(GLIBC_2.3)[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcstombs_chk(GLIBC_2.4)</td>
<td>inet_ntoa[SUSv4]</td>
<td>strcasestr[LSB]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcstoul_internal</td>
<td>inet_ntop[SUSv4]</td>
<td>stremat[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>__wcstomb_chk(GLIBC_2.4)</td>
<td>inet_pton[SUSv4]</td>
<td>strchr[SUSv4]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Prototype</td>
<td>Notes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------</td>
<td>----------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__wmemcpy_chk(GLIBC_2.4)[LSB]</td>
<td>initgroups[LSB]</td>
<td>streq[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__wmemmove_chk(GLIBC_2.4)[LSB]</td>
<td>initstate[SV4]</td>
<td>strcoll[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__wmemcpy_chk(GLIBC_2.4)[LSB]</td>
<td>initstate_r[LSB]</td>
<td>strcoll_l(GLIBC_2.3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__wmemset_chk(GLIBC_2.4)</td>
<td>inotify_add_watch(GLIBC_2.4)[LSB]</td>
<td>strcpy[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__wprintf_chk(GLIBC_2.4)</td>
<td>inotify_init(GLIBC_2.4)[LSB]</td>
<td>strcspsn[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__wmemcpy_chk(GLIBC_2.4)</td>
<td>inotify_rm_watch(GLIBC_2.4)[LSB]</td>
<td>strlen[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__wmemmove_chk(GLIBC_2.4)</td>
<td>insque[SV4]</td>
<td>streqerror[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__wmempcpy_chk(GLIBC_2.4)</td>
<td>ioctl[LSB]</td>
<td>strerr_l[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__wprintf_chk(GLIBC_2.4)</td>
<td>isalnum[SV4]</td>
<td>strerr_l_r[LSB]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__wprintf_chk(GLIBC_2.4)</td>
<td>isalnum_l(GLIBC_2.3)</td>
<td>strfmon[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__xpg_basename[LSB]</td>
<td>isalpha[SV4]</td>
<td>strfmon_l(GLIBC_2.3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__xpg_sigpause[LSB]</td>
<td>isalpha_l(GLIBC_2.3)</td>
<td>strftm[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__xpg_strerror_r(GLIBC_2.3.4)</td>
<td>isascii[SV4]</td>
<td>strfmon_l_r[LSB]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>__xstat64[LSB]</td>
<td>isdigit_l(GLIBC_2.3)</td>
<td>strncasecmp[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>_longjmp[SV4]</td>
<td>isgraph_l(GLIBC_2.3)</td>
<td>strncasecmp_l_r[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>_toupper[SV4]</td>
<td>islower_l(GLIBC_2.3)</td>
<td>strnclmp[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>alphasort64[LSB]</td>
<td>isprint_l(GLIBC_2.3)</td>
<td>strndup[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>argz_add_sep[LSB]</td>
<td>isgraph_l(GLIBC_2.3)</td>
<td>strpbrk[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>adjtime[LSB]</td>
<td>islower_l(GLIBC_2.3)</td>
<td>strtof[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>alphasort64[LSB]</td>
<td>isprint_l(GLIBC_2.3)</td>
<td>strprof[SV4]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
<table>
<thead>
<tr>
<th>Function</th>
<th>Library</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>argz_append</td>
<td>LSB</td>
<td>Add an element to a list</td>
</tr>
<tr>
<td>argz_count</td>
<td>LSB</td>
<td>Count the elements in a list</td>
</tr>
<tr>
<td>argz_create</td>
<td>LSB</td>
<td>Create a list</td>
</tr>
<tr>
<td>argz_create_sep</td>
<td>LSB</td>
<td>Create a list with separators</td>
</tr>
<tr>
<td>argz_delete</td>
<td>LSB</td>
<td>Delete an element from a list</td>
</tr>
<tr>
<td>argz_extract</td>
<td>LSB</td>
<td>Extract an element from a list</td>
</tr>
<tr>
<td>argz_insert</td>
<td>LSB</td>
<td>Insert an element into a list</td>
</tr>
<tr>
<td>argz_next</td>
<td>LSB</td>
<td>Get the next element in a list</td>
</tr>
<tr>
<td>asctime</td>
<td>SUSv4</td>
<td>Return the system time</td>
</tr>
<tr>
<td>asprintf</td>
<td>LSB</td>
<td>Format a string</td>
</tr>
<tr>
<td>atof</td>
<td>SUSv4</td>
<td>Convert a string to a number</td>
</tr>
<tr>
<td>atoi</td>
<td>SUSv4</td>
<td>Convert a string to a number</td>
</tr>
<tr>
<td>atoll</td>
<td>SUSv4</td>
<td>Convert a string to a number</td>
</tr>
<tr>
<td>bcmp</td>
<td>SUSv3</td>
<td>Compare two blocks of memory</td>
</tr>
<tr>
<td>bcopy</td>
<td>SUSv3</td>
<td>Copy memory</td>
</tr>
<tr>
<td>basename</td>
<td>SUSv4</td>
<td>Return the base of a path</td>
</tr>
<tr>
<td>bsd_signal</td>
<td>SUSv3</td>
<td>Signal a process</td>
</tr>
<tr>
<td>bind</td>
<td>SUSv4</td>
<td>Bind a socket to an address</td>
</tr>
<tr>
<td>bind_textdomain_codeset</td>
<td>LSB</td>
<td>Bind a text domain code set</td>
</tr>
<tr>
<td>bindresvport</td>
<td>LSB</td>
<td>Bind a reserver port</td>
</tr>
<tr>
<td>bndtextdomain</td>
<td>LSB</td>
<td>Bind a text domain</td>
</tr>
<tr>
<td>brk</td>
<td>SUSv2</td>
<td>Break a process</td>
</tr>
<tr>
<td>svc_getreqset</td>
<td>SVID.4</td>
<td>Get a request set</td>
</tr>
<tr>
<td>svc_register</td>
<td>LSB</td>
<td>Register a service</td>
</tr>
<tr>
<td>svc_run</td>
<td>LSB</td>
<td>Run a service</td>
</tr>
<tr>
<td>svcerr_auth</td>
<td>SVID.4</td>
<td>Error code for auth failure</td>
</tr>
<tr>
<td>svcerr_decode</td>
<td>SVID.4</td>
<td>Decode an error code</td>
</tr>
<tr>
<td>svcerr_noproc</td>
<td>SVID.4</td>
<td>Error code for no process</td>
</tr>
<tr>
<td>svcerr_systemerr</td>
<td>SVID.4</td>
<td>System error code</td>
</tr>
<tr>
<td>svcerr_weakauth</td>
<td>SVID.4</td>
<td>Weak authorization error</td>
</tr>
<tr>
<td>isalnum</td>
<td>SUSv4</td>
<td>Check if a string is alphanumerical</td>
</tr>
<tr>
<td>isalpha</td>
<td>SUSv4</td>
<td>Check if a string is alphabetical</td>
</tr>
<tr>
<td>isblank</td>
<td>SUSv4</td>
<td>Check if a character is blank</td>
</tr>
<tr>
<td>iscntrl</td>
<td>SUSv4</td>
<td>Check if a character is a control character</td>
</tr>
<tr>
<td>isdigit</td>
<td>SUSv4</td>
<td>Check if a character is a digit</td>
</tr>
<tr>
<td>isdigit_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a digit (locale sensitive)</td>
</tr>
<tr>
<td>isgraph</td>
<td>SUSv4</td>
<td>Check if a character is a graphic</td>
</tr>
<tr>
<td>isgraph_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a graphic (locale sensitive)</td>
</tr>
<tr>
<td>islower</td>
<td>SUSv4</td>
<td>Check if a character is lowercase</td>
</tr>
<tr>
<td>islower_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is lowercase (locale sensitive)</td>
</tr>
<tr>
<td>isprint</td>
<td>SUSv4</td>
<td>Check if a character is printable</td>
</tr>
<tr>
<td>isprint_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is printable (locale sensitive)</td>
</tr>
<tr>
<td>ispunct</td>
<td>SUSv4</td>
<td>Check if a character is a punctuation</td>
</tr>
<tr>
<td>ispunct_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a punctuation (locale sensitive)</td>
</tr>
<tr>
<td>isxdigit</td>
<td>SUSv4</td>
<td>Check if a character is a hexadecimal digit</td>
</tr>
<tr>
<td>isxdigit_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a hexadecimal digit (locale sensitive)</td>
</tr>
<tr>
<td>isxdigit</td>
<td>SUSv4</td>
<td>Check if a character is a hexadecimal digit</td>
</tr>
<tr>
<td>isxdigit_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a hexadecimal digit (locale sensitive)</td>
</tr>
<tr>
<td>iswalpha</td>
<td>SUSv4</td>
<td>Check if a character is a wide alphabet</td>
</tr>
<tr>
<td>iswalpha_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a wide alphabet (locale sensitive)</td>
</tr>
<tr>
<td>iswalnum</td>
<td>SUSv4</td>
<td>Check if a character is a wide alphanumeric</td>
</tr>
<tr>
<td>iswalnum_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a wide alphanumeric (locale sensitive)</td>
</tr>
<tr>
<td>iswctype</td>
<td>SUSv4</td>
<td>Check if a character is a wide character type</td>
</tr>
<tr>
<td>iswctype_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a wide character type (locale sensitive)</td>
</tr>
<tr>
<td>iswdigit</td>
<td>SUSv4</td>
<td>Check if a character is a wide digit</td>
</tr>
<tr>
<td>iswdigit_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a wide digit (locale sensitive)</td>
</tr>
<tr>
<td>iswxdigit</td>
<td>SUSv4</td>
<td>Check if a character is a wide hexadecimal digit</td>
</tr>
<tr>
<td>iswxdigit_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a wide hexadecimal digit (locale sensitive)</td>
</tr>
<tr>
<td>iswprint</td>
<td>SUSv4</td>
<td>Check if a character is a wide printable</td>
</tr>
<tr>
<td>iswprint_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a wide printable (locale sensitive)</td>
</tr>
<tr>
<td>iswspace</td>
<td>SUSv4</td>
<td>Check if a character is a wide space</td>
</tr>
<tr>
<td>iswspace_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a wide space (locale sensitive)</td>
</tr>
<tr>
<td>iswgraph</td>
<td>SUSv4</td>
<td>Check if a character is a wide graphic</td>
</tr>
<tr>
<td>iswgraph_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a wide graphic (locale sensitive)</td>
</tr>
<tr>
<td>iswupper</td>
<td>SUSv4</td>
<td>Check if a character is a wide upper case</td>
</tr>
<tr>
<td>iswupper_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a wide upper case (locale sensitive)</td>
</tr>
<tr>
<td>iswucase</td>
<td>SUSv4</td>
<td>Check if a character is a wide uppercase</td>
</tr>
<tr>
<td>iswucase_l</td>
<td>GLIBC_2.3</td>
<td>Check if a character is a wide uppercase (locale sensitive)</td>
</tr>
<tr>
<td>svctcp_create</td>
<td>LSB</td>
<td>Create a TCP socket</td>
</tr>
<tr>
<td>svcudp_create</td>
<td>LSB</td>
<td>Create a UDP socket</td>
</tr>
<tr>
<td>strtof</td>
<td>SUSv4</td>
<td>Convert a string to a float</td>
</tr>
<tr>
<td>strtok</td>
<td>SUSv4</td>
<td>Split a string by delimiters</td>
</tr>
<tr>
<td>strtok_r</td>
<td>SUSv4</td>
<td>Split a string by delimiters</td>
</tr>
<tr>
<td>strtol</td>
<td>SUSv4</td>
<td>Convert a string to an long</td>
</tr>
<tr>
<td>strtoimax</td>
<td>SUSv4</td>
<td>Convert a string to an integer</td>
</tr>
<tr>
<td>strtoql</td>
<td>SUSv4</td>
<td>Convert a string to a long unsigned integer</td>
</tr>
<tr>
<td>strtoql</td>
<td>SUSv4</td>
<td>Convert a string to a long unsigned integer</td>
</tr>
<tr>
<td>strtoq</td>
<td>SUSv4</td>
<td>Convert a string to a qword</td>
</tr>
<tr>
<td>strxfrm</td>
<td>SUSv4</td>
<td>Convert a string to a case insensitive</td>
</tr>
<tr>
<td>strxfrm_l</td>
<td>GLIBC_2.3</td>
<td>Convert a string to a case insensitive (locale sensitive)</td>
</tr>
<tr>
<td>svcerr_auth</td>
<td>SVID.4</td>
<td>Error code for auth failure</td>
</tr>
<tr>
<td>svcerr_decode</td>
<td>SVID.4</td>
<td>Decode an error code</td>
</tr>
<tr>
<td>svcerr_noproc</td>
<td>SVID.4</td>
<td>Error code for no process</td>
</tr>
<tr>
<td>svcerr_systemerr</td>
<td>SVID.4</td>
<td>System error code</td>
</tr>
<tr>
<td>svcerr_weakauth</td>
<td>SVID.4</td>
<td>Weak authorization error</td>
</tr>
<tr>
<td>svc_getreqset</td>
<td>SVID.4</td>
<td>Get a request set</td>
</tr>
<tr>
<td>svc_register</td>
<td>LSB</td>
<td>Register a service</td>
</tr>
<tr>
<td>svc_run</td>
<td>LSB</td>
<td>Run a service</td>
</tr>
<tr>
<td>svcerr_auth</td>
<td>SVID.4</td>
<td>Error code for auth failure</td>
</tr>
<tr>
<td>svcerr_decode</td>
<td>SVID.4</td>
<td>Decode an error code</td>
</tr>
<tr>
<td>svcerr_noproc</td>
<td>SVID.4</td>
<td>Error code for no process</td>
</tr>
<tr>
<td>svcerr_systemerr</td>
<td>SVID.4</td>
<td>System error code</td>
</tr>
<tr>
<td>svcerr_weakauth</td>
<td>SVID.4</td>
<td>Weak authorization error</td>
</tr>
<tr>
<td>svc_getreqset</td>
<td>SVID.4</td>
<td>Get a request set</td>
</tr>
<tr>
<td>Function</td>
<td>Standard</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>bsearch</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>isxdigit_l(GLIBC_2.3)</td>
<td>SUSv3</td>
<td></td>
</tr>
<tr>
<td>swapcontext</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>brutoe</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>jrand48</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>swprintf</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>bzero</td>
<td>SUSv3</td>
<td></td>
</tr>
<tr>
<td>jrand48_r</td>
<td>LSB</td>
<td></td>
</tr>
<tr>
<td>swscanf</td>
<td>LSB</td>
<td></td>
</tr>
<tr>
<td>calloc</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>key_decryptsession</td>
<td>SVID_4</td>
<td></td>
</tr>
<tr>
<td>symlink</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>callrpc</td>
<td>RPC + XDR</td>
<td></td>
</tr>
<tr>
<td>kill</td>
<td>LSB</td>
<td></td>
</tr>
<tr>
<td>symlinkat</td>
<td>GLIBC_2.4</td>
<td></td>
</tr>
<tr>
<td>catclose</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>killpg</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>catopen</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>labs</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>syslog</td>
<td>LSB</td>
<td></td>
</tr>
<tr>
<td>cfgetispeed</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>lchown</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>cfgetospeed</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>lcong48</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>cfmakeraw</td>
<td>LSB</td>
<td></td>
</tr>
<tr>
<td>lcong48_r</td>
<td>LSB</td>
<td></td>
</tr>
<tr>
<td>cfsetispeed</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>ldiv</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>cfsetospeed</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>llabs</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>cfsetspeed</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>tclflush</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>chdir</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>linkat</td>
<td>GLIBC_2.4</td>
<td></td>
</tr>
<tr>
<td>chmod</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>listen</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>chown</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>llabs</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>chroot</td>
<td>SUSv2</td>
<td></td>
</tr>
<tr>
<td>lldiv</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clearerr</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>localeconv</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clearerr_unlocked</td>
<td>LSB</td>
<td></td>
</tr>
<tr>
<td>localtime</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clnt_create</td>
<td>SVID_4</td>
<td></td>
</tr>
<tr>
<td>localtime_r</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clnt_pcreateerror</td>
<td>SVID_4</td>
<td></td>
</tr>
<tr>
<td>lock</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clnt_perrorno</td>
<td>SVID_4</td>
<td></td>
</tr>
<tr>
<td>lockf64</td>
<td>LFS</td>
<td></td>
</tr>
<tr>
<td>clnt_perror</td>
<td>SVID_4</td>
<td></td>
</tr>
<tr>
<td>longjmp</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clnt_perrorno</td>
<td>SVID_4</td>
<td></td>
</tr>
<tr>
<td>lrand48</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>time</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clnt_sperror</td>
<td>SVID_4</td>
<td></td>
</tr>
<tr>
<td>lrand48_r</td>
<td>LSB</td>
<td></td>
</tr>
<tr>
<td>times</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clnt_sperror</td>
<td>SVID_4</td>
<td></td>
</tr>
<tr>
<td>lsearch</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>tmpfile</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clntraw_create</td>
<td>RPC + XDR</td>
<td></td>
</tr>
<tr>
<td>lseek</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>tmpfile64</td>
<td>LFS</td>
<td></td>
</tr>
<tr>
<td>clnttcp_create</td>
<td>RPC + XDR</td>
<td></td>
</tr>
<tr>
<td>lseek64</td>
<td>LFS</td>
<td></td>
</tr>
<tr>
<td>tmpnam</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clntudp_bufcreate</td>
<td>RPC + XDR</td>
<td></td>
</tr>
<tr>
<td>lutimes</td>
<td>GLIBC_2.3</td>
<td></td>
</tr>
<tr>
<td>toascii</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clntudp_create</td>
<td>RPC + XDR</td>
<td></td>
</tr>
<tr>
<td>malloc</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>tolerant</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>clock</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>mblen</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>toupper</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>closedir</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>mbrlen</td>
<td>SUSv4</td>
<td></td>
</tr>
<tr>
<td>toupper_l(GLIBC_2.3)</td>
<td>SUSv4</td>
<td></td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
<table>
<thead>
<tr>
<th>Function</th>
<th>Lib</th>
<th>Function</th>
<th>Lib</th>
<th>Function</th>
<th>Lib</th>
</tr>
</thead>
<tbody>
<tr>
<td>closelog</td>
<td>SUSv4</td>
<td>mbtowc</td>
<td>SUSv4</td>
<td>towchar</td>
<td>SUSv4</td>
</tr>
<tr>
<td>confstr</td>
<td>SUSv4</td>
<td>mbsinit</td>
<td>SUSv4</td>
<td>towchar_r</td>
<td>GLIBC_2.3</td>
</tr>
<tr>
<td>connect</td>
<td>SUSv4</td>
<td>mbsrtowcs</td>
<td>SUSv4</td>
<td>towchar</td>
<td>SUSv4</td>
</tr>
<tr>
<td>creat</td>
<td>SUSv4</td>
<td>mbsrtowcs</td>
<td>SUSv4</td>
<td>towchar_r</td>
<td>GLIBC_2.3</td>
</tr>
<tr>
<td>creat64</td>
<td>LFS</td>
<td>mbsrtowcs</td>
<td>SUSv4</td>
<td>towchar</td>
<td>SUSv4</td>
</tr>
<tr>
<td>ctermid</td>
<td>SUSv4</td>
<td>mbtowc</td>
<td>SUSv4</td>
<td>towchar_r</td>
<td>GLIBC_2.3</td>
</tr>
<tr>
<td>ctime</td>
<td>SUSv4</td>
<td>memccpy</td>
<td>SUSv4</td>
<td>truncate</td>
<td>SUSv4</td>
</tr>
<tr>
<td>ctime_r</td>
<td>SUSv4</td>
<td>memchr</td>
<td>SUSv4</td>
<td>truncate64</td>
<td>LFS</td>
</tr>
<tr>
<td>cuserid</td>
<td>SUSv2</td>
<td>memcmp</td>
<td>SUSv4</td>
<td>tsearch</td>
<td>SUSv4</td>
</tr>
<tr>
<td>daemon</td>
<td>LSB</td>
<td>memcpyp</td>
<td>SUSv4</td>
<td>ttyname</td>
<td>SUSv4</td>
</tr>
<tr>
<td>dcegettext</td>
<td>LSB</td>
<td>memmove</td>
<td>SUSv4</td>
<td>twalk</td>
<td>SUSv4</td>
</tr>
<tr>
<td>dgettext</td>
<td>LSB</td>
<td>memchr</td>
<td>LSB</td>
<td>tzset</td>
<td>SUSv4</td>
</tr>
<tr>
<td>difftime</td>
<td>SUSv4</td>
<td>memset</td>
<td>SUSv4</td>
<td>ularm</td>
<td>SUSv3</td>
</tr>
<tr>
<td>dirfd</td>
<td>SUSv4</td>
<td>mkdircmp</td>
<td>SUSv4</td>
<td>ulimit</td>
<td>SUSv4</td>
</tr>
<tr>
<td>dirname</td>
<td>SUSv4</td>
<td>mkdirat</td>
<td>GLIBC_2.4</td>
<td>umask</td>
<td>SUSv4</td>
</tr>
<tr>
<td>div</td>
<td>SUSv4</td>
<td>mktemp</td>
<td>SUSv4</td>
<td>uname</td>
<td>SUSv4</td>
</tr>
<tr>
<td>dl_iterate_phdr</td>
<td>LSB</td>
<td>mkfifo</td>
<td>SUSv4</td>
<td>unget</td>
<td>SUSv4</td>
</tr>
<tr>
<td>dngettext</td>
<td>LSB</td>
<td>mkfifoat</td>
<td>GLIBC_2.4</td>
<td>ungetwc</td>
<td>SUSv4</td>
</tr>
<tr>
<td>dprintf</td>
<td>SUSv4</td>
<td>mkstemp</td>
<td>SUSv4</td>
<td>unlink</td>
<td>LSB</td>
</tr>
<tr>
<td>drand48</td>
<td>SUSv4</td>
<td>mkstemp64</td>
<td>LSB</td>
<td>unlinkat</td>
<td>GLIBC_2.4</td>
</tr>
<tr>
<td>drand48_r</td>
<td>LSB</td>
<td>mktemp</td>
<td>SUSv3</td>
<td>unlockpt</td>
<td>SUSv4</td>
</tr>
<tr>
<td>dup</td>
<td>SUSv4</td>
<td>mktime</td>
<td>SUSv4</td>
<td>unsetenv</td>
<td>SUSv4</td>
</tr>
<tr>
<td>dup2</td>
<td>SUSv4</td>
<td>mlock</td>
<td>SUSv4</td>
<td>uselocale</td>
<td>GLIBC_2.3</td>
</tr>
<tr>
<td>duplocale</td>
<td>GLIBC_2.3</td>
<td>mlockall</td>
<td>SUSv4</td>
<td>usleep</td>
<td>SUSv3</td>
</tr>
<tr>
<td>duplocale</td>
<td>SUSv4</td>
<td>ecvt</td>
<td>SUSv3</td>
<td>utime</td>
<td>SUSv4</td>
</tr>
<tr>
<td>endgrent</td>
<td>SUSv4</td>
<td>mmap</td>
<td>SUSv4</td>
<td>utimensat</td>
<td>GLIBC_2.6</td>
</tr>
<tr>
<td>endprotosent</td>
<td>SUSv4</td>
<td>mmap64</td>
<td>LFS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>endpwent</td>
<td>SUSv4</td>
<td>mrand48</td>
<td>SUSv4</td>
<td>utmpname</td>
<td>LSB</td>
</tr>
<tr>
<td>endservent</td>
<td>SUSv4</td>
<td>mrand48_r</td>
<td>LSB</td>
<td>vasprintf</td>
<td>LSB</td>
</tr>
<tr>
<td>endutent</td>
<td>LSB</td>
<td>memmapi</td>
<td>LSB</td>
<td>vdprintf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>endutentx</td>
<td>SUSv4</td>
<td>msgctl</td>
<td>SUSv4</td>
<td>vfork</td>
<td>SUSv3</td>
</tr>
<tr>
<td>envz_add</td>
<td>LSB</td>
<td>msgget</td>
<td>SUSv4</td>
<td>vfork</td>
<td>SUSv4</td>
</tr>
<tr>
<td>envz_entry</td>
<td>LSB</td>
<td>msgrecv</td>
<td>SUSv4</td>
<td>vprintf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>envz_get</td>
<td>LSB</td>
<td>msgsnd</td>
<td>SUSv4</td>
<td>vscanf</td>
<td>LSB</td>
</tr>
<tr>
<td>envz_merge</td>
<td>LSB</td>
<td>msync</td>
<td>SUSv4</td>
<td>vfwprintf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>Function</td>
<td>Library</td>
<td>Library</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------</td>
<td>---------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>envz_remove[LSB]</td>
<td>munlock[SUSv4]</td>
<td>vfwscanf[LSB]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>epoll_create(GLIBC_2.3.2)[LSB]</td>
<td>munmap[SUSv4]</td>
<td>vscanf[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>epoll_ctl(GLIBC_2.3.2)[LSB]</td>
<td>nanosleep[SUSv4]</td>
<td>vsprintf[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>epoll_wait(GLIBC_2.3.2)[LSB]</td>
<td>newlocale(GLIBC_2.3)[SUSv4]</td>
<td>vsscanf[LSB]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>erand48_r[LSB]</td>
<td>nftw64[LFS]</td>
<td>vswscanf[LSB]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>error[LSB]</td>
<td>ngettext[LSB]</td>
<td>vwscanf[LSB]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>faccessat(GLIBC_2.4)[SUSv4]</td>
<td>open_wmemstream(GLIBC_2.4)[SUSv4]</td>
<td>wcpcpy[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fchmod[SUSv4]</td>
<td>openat64(GLIBC_2.4)[LSB]</td>
<td>wctomb[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fdopendir(GLIBC_2.4)[SUSv4]</td>
<td>pmap_set[LSB]</td>
<td>wcscpy[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fexecve[SUSv4]</td>
<td>posix_fadvise64[LSB]</td>
<td>wcscasecmp_l(GLIBC_2.3)[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Library</td>
<td>Function</td>
<td>Library</td>
<td>Function</td>
<td>Library</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------</td>
<td>-------------------------------</td>
<td>-----------------------</td>
<td>-------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>fflush_unlocked[LSB]</td>
<td></td>
<td>posix_fallocate64[LSB]</td>
<td></td>
<td>wcsncmp[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>fgetpos64[LFS]</td>
<td></td>
<td>posix_spawn_file_actions</td>
<td></td>
<td>wcsrchr[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>fgets_unlocked[LSB]</td>
<td></td>
<td>posix_spawn_file_actions</td>
<td></td>
<td>wcsspn[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>fgetwc_unlocked[LSB]</td>
<td></td>
<td>posix_spawn_file_actions</td>
<td></td>
<td>wcstod[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>fgetws_unlocked[LSB]</td>
<td></td>
<td>posix_spawn_file_actions</td>
<td></td>
<td>wcstoimax[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>fileno[LSB]</td>
<td></td>
<td>posix_spawnattr_getflags</td>
<td></td>
<td>wcstok[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>fileno_unlocked[LSB]</td>
<td></td>
<td>posix_spawnattr_getpgroup</td>
<td></td>
<td>wcstoimax[SUSv4]</td>
<td></td>
</tr>
<tr>
<td>flock[LSB]</td>
<td></td>
<td>posix_spawnattr_getpolicy</td>
<td></td>
<td>wcstok[LSP]</td>
<td></td>
</tr>
<tr>
<td>flockfile[SUSv4]</td>
<td></td>
<td>posix_spawnattr_getsigdeath</td>
<td></td>
<td>wcstoul[LSP]</td>
<td></td>
</tr>
<tr>
<td>fmemopen[SUSv4]</td>
<td></td>
<td>posix_spawnattr_getsigmask</td>
<td></td>
<td>wcstoul[LSP]</td>
<td></td>
</tr>
<tr>
<td>fnmsgmsg[SUSv4]</td>
<td></td>
<td>posix_spawnattr_getinit</td>
<td></td>
<td>wcstoull[LSP]</td>
<td></td>
</tr>
<tr>
<td>fnmatch[LSB]</td>
<td></td>
<td>posix_spawnattr_setflags</td>
<td></td>
<td>wcstoull[LSP]</td>
<td></td>
</tr>
<tr>
<td>fopen[SUSv4]</td>
<td></td>
<td>posix_spawnattr_setpgroup</td>
<td></td>
<td>wcstoull[LSP]</td>
<td></td>
</tr>
<tr>
<td>fopen64[LFS]</td>
<td></td>
<td>posix_spawnattr_setsched</td>
<td></td>
<td>wcstoull[LSP]</td>
<td></td>
</tr>
<tr>
<td>fork[SUSv4]</td>
<td></td>
<td>posix_spawnattr_setsched</td>
<td></td>
<td>wcstoull[LSP]</td>
<td></td>
</tr>
<tr>
<td>fpathconf[SUSv4]</td>
<td></td>
<td>posix_spawnattr_setsigdeath</td>
<td></td>
<td>wcs wcs [SUSv3]</td>
<td></td>
</tr>
<tr>
<td>fprintf[SUSv4]</td>
<td></td>
<td>posix_spawnattr_setsigmask</td>
<td></td>
<td>wcstoull[LSP]</td>
<td></td>
</tr>
<tr>
<td>fputc_unlocked[LSB]</td>
<td></td>
<td>pread[SUSv4]</td>
<td></td>
<td>wcsxfm [GLIBC_2.3]</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Library</td>
<td>Description</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------</td>
<td>--------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>puts_unlocked</td>
<td>LSB</td>
<td>wctomb[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pselect</td>
<td>SUSv4</td>
<td>wctrans[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>psiginfo(GLIBC_2.10)</td>
<td>SUSv4</td>
<td>wctrans_l(GLIBC_2.3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>psignal</td>
<td>SUSv4</td>
<td>wctype[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>psiginfo(GLIBC_2.3)</td>
<td>SUSv4</td>
<td>wctype_l(GLIBC_2.3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ptrace</td>
<td>LSB</td>
<td>wmemmove[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>psignal</td>
<td>SUSv4</td>
<td>wmemmove[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>putenv</td>
<td>SUSv4</td>
<td>wmemset[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>putwc</td>
<td>SUSv4</td>
<td>xdr_array[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>putwc_unlocked</td>
<td>LSB</td>
<td>xdr_bool[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>putwc_unlocked</td>
<td>SUSv4</td>
<td>xdr_callhdr[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>putwchar</td>
<td>SUSv4</td>
<td>xdr_callmsg[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>putwchar_unlocked</td>
<td>LSB</td>
<td>xdr_accepted_reply[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>putwc</td>
<td>SUSv4</td>
<td>xdr_rejected_reply[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>putwchar</td>
<td>SUSv4</td>
<td>xdr_replymsg[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>read</td>
<td>SUSv4</td>
<td>xdr_char[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>read64</td>
<td>LFS</td>
<td>xdr_double[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>random</td>
<td>SUSv4</td>
<td>xdr_enum[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>random_r</td>
<td>LSB</td>
<td>xdr_float[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir</td>
<td>SUSv4</td>
<td>xdr_free[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>LFS</td>
<td>xdr_free[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>SUSv4</td>
<td>xdr_free[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>LSB</td>
<td>xdr_long[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>SUSv4</td>
<td>xdr_opaque[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>LFS</td>
<td>xdr_opaque[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>SUSv4</td>
<td>xdr_opaque[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>LSB</td>
<td>xdr_rejected_reply[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>SUSv4</td>
<td>xdr_rejected_reply[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>LFS</td>
<td>xdr_repliedmsg[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>SUSv4</td>
<td>xdr_repliedmsg[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>LSB</td>
<td>xdr_repliedmsg[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>SUSv4</td>
<td>xdr_repliedmsg[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>LFS</td>
<td>xdr_repliedmsg[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>readdir64</td>
<td>SUSv4</td>
<td>xdr_repliedmsg[SVID.4]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Annex A Alphabetical Listing of Interfaces by Library

A.2 libcrypt

The behavior of the interfaces in this library is specified by the following Standards.

This Specification [LSB]
POSIX 1003.1-2008 (ISO/IEC 9945-2009) [SUSv4]

Table A-3 libcrypt Function Interfaces

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>crypt_r[LSB]</td>
<td>encrypt_r[LSB]</td>
<td>setkey_r[LSB]</td>
</tr>
</tbody>
</table>

A.3 libdl

The behavior of the interfaces in this library is specified by the following Standards.

This Specification [LSB]
POSIX 1003.1-2008 (ISO/IEC 9945-2009) [SUSv4]

Table A-4 libdl Function Interfaces

<table>
<thead>
<tr>
<th>dladdr[LSB]</th>
<th>dlerror[SUSv4]</th>
<th>dlvsym[LSB]</th>
</tr>
</thead>
</table>

A.4 libgcc_s

The behavior of the interfaces in this library is specified by the following Standards.

This Specification [LSB]
A.5 libm

The behavior of the interfaces in this library is specified by the following Standards.

This Specification [LSB]
POSIX 1003.1-2001 (ISO/IEC 9945-2003) [SUSv3]
POSIX 1003.1-2008 (ISO/IEC 9945-2009) [SUSv4]
<table>
<thead>
<tr>
<th>Function</th>
<th>Library</th>
<th>Function</th>
<th>Library</th>
<th>Function</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>atanl</td>
<td>SUSv4</td>
<td>expm1</td>
<td>SUSv4</td>
<td>nan</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cabs</td>
<td>SUSv4</td>
<td>expm1</td>
<td>SUSv4</td>
<td>nanf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cabsf</td>
<td>SUSv4</td>
<td>expm1</td>
<td>SUSv4</td>
<td>nanl</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cabsl</td>
<td>SUSv4</td>
<td>fabs</td>
<td>SUSv4</td>
<td>nearbyint</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cacos</td>
<td>SUSv4</td>
<td>fabs</td>
<td>SUSv4</td>
<td>nearbyintf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cacosf</td>
<td>SUSv4</td>
<td>fabs</td>
<td>SUSv4</td>
<td>nearbyintl</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cacosfl</td>
<td>SUSv4</td>
<td>fdim</td>
<td>SUSv4</td>
<td>nextafter</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cacosf</td>
<td>SUSv4</td>
<td>fdimf</td>
<td>SUSv4</td>
<td>nextafterf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cacosf</td>
<td>SUSv4</td>
<td>fdiml</td>
<td>SUSv4</td>
<td>nextafterl</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cacos</td>
<td>SUSv4</td>
<td>feclearexcept</td>
<td>SUSv4</td>
<td>nexttoward</td>
<td>SUSv4</td>
</tr>
<tr>
<td>carg</td>
<td>SUSv4</td>
<td>fedisableexcept</td>
<td>LSB</td>
<td>nexttowardf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cargf</td>
<td>SUSv4</td>
<td>feenableexcept</td>
<td>LSB</td>
<td>nexttowardf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cargfl</td>
<td>SUSv4</td>
<td>fegetenv</td>
<td>SUSv4</td>
<td>nexttowardfl</td>
<td>SUSv4</td>
</tr>
<tr>
<td>casin</td>
<td>SUSv4</td>
<td>fegetexcept</td>
<td>LSB</td>
<td>pow</td>
<td>SUSv4</td>
</tr>
<tr>
<td>casinf</td>
<td>SUSv4</td>
<td>fegetexceptf</td>
<td>SUSv4</td>
<td>pow10</td>
<td>LSB</td>
</tr>
<tr>
<td>casinh</td>
<td>SUSv4</td>
<td>fegetround</td>
<td>SUSv4</td>
<td>pow10</td>
<td>LSB</td>
</tr>
<tr>
<td>casinhf</td>
<td>SUSv4</td>
<td>feholdexcept</td>
<td>SUSv4</td>
<td>powf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>casinhf</td>
<td>SUSv4</td>
<td>feraiseexcept</td>
<td>SUSv4</td>
<td>powl</td>
<td>SUSv4</td>
</tr>
<tr>
<td>casinl</td>
<td>SUSv4</td>
<td>fetenv</td>
<td>SUSv4</td>
<td>remainder</td>
<td>SUSv4</td>
</tr>
<tr>
<td>catan</td>
<td>SUSv4</td>
<td>fetextexcept</td>
<td>SUSv4</td>
<td>remainderf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>catanl</td>
<td>SUSv4</td>
<td>fetextexceptl</td>
<td>SUSv4</td>
<td>remainderfl</td>
<td>SUSv4</td>
</tr>
<tr>
<td>catanh</td>
<td>SUSv4</td>
<td>feupdateenv</td>
<td>SUSv4</td>
<td>remquof</td>
<td>SUSv4</td>
</tr>
<tr>
<td>catanhf</td>
<td>SUSv4</td>
<td>feupdateenvf</td>
<td>SUSv4</td>
<td>remquof</td>
<td>SUSv4</td>
</tr>
<tr>
<td>catanhl</td>
<td>SUSv4</td>
<td>finite</td>
<td>LSB</td>
<td>remquo</td>
<td>SUSv4</td>
</tr>
<tr>
<td>catanl</td>
<td>SUSv4</td>
<td>finitef</td>
<td>LSB</td>
<td>rint</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cbrtf</td>
<td>SUSv4</td>
<td>floor</td>
<td>SUSv4</td>
<td>rint</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cbrtf</td>
<td>SUSv4</td>
<td>floorf</td>
<td>SUSv4</td>
<td>round</td>
<td>SUSv4</td>
</tr>
<tr>
<td>ccos</td>
<td>SUSv4</td>
<td>floorf</td>
<td>SUSv4</td>
<td>roundf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>ccosf</td>
<td>SUSv4</td>
<td>fma</td>
<td>SUSv4</td>
<td>roundf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>ccosfl</td>
<td>SUSv4</td>
<td>fmaf</td>
<td>SUSv4</td>
<td>scalbf</td>
<td>LSB</td>
</tr>
<tr>
<td>ccosf</td>
<td>SUSv4</td>
<td>fmal</td>
<td>SUSv4</td>
<td>scalbl</td>
<td>LSB</td>
</tr>
<tr>
<td>ccosl</td>
<td>SUSv4</td>
<td>fmax</td>
<td>SUSv4</td>
<td>scalblf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>ccosl</td>
<td>SUSv4</td>
<td>fmaxf</td>
<td>SUSv4</td>
<td>scalblnf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cexpf</td>
<td>SUSv4</td>
<td>fmin</td>
<td>SUSv4</td>
<td>scalbn</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cexpfl</td>
<td>SUSv4</td>
<td>fminf</td>
<td>SUSv4</td>
<td>scalbnf</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cexpf</td>
<td>SUSv4</td>
<td>fmod</td>
<td>SUSv4</td>
<td>scalbn</td>
<td>SUSv4</td>
</tr>
<tr>
<td>cexpfl</td>
<td>SUSv4</td>
<td>fmodf</td>
<td>SUSv4</td>
<td>significand</td>
<td>LSB</td>
</tr>
<tr>
<td>cimag</td>
<td>SUSv4</td>
<td>fmodl</td>
<td>SUSv4</td>
<td>significandf</td>
<td>LSB</td>
</tr>
<tr>
<td>cimag</td>
<td>SUSv4</td>
<td>frexp</td>
<td>SUSv4</td>
<td>significand</td>
<td>LSB</td>
</tr>
<tr>
<td>cimag</td>
<td>SUSv4</td>
<td>frexpf</td>
<td>SUSv4</td>
<td>sin</td>
<td>SUSv4</td>
</tr>
</tbody>
</table>
Table A-7 libm Data Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>signgam[SUSv4]</td>
<td>SUSv4</td>
</tr>
</tbody>
</table>

A.6 libncurses

The behavior of the interfaces in this library is specified by the following Standards.

This Specification [LSB]
X/Open Curses, Issue 7 [X-CURSES]

Table A-8 libncurses Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>addch[X-CURSES]</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>mvdelch[X-CURSES]</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>slk_refresh[X-CURSES]</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>addchnstr[X-CURSES]</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>mvderwin[X-CURSES]</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>slk_restore[X-CURSES]</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>addchstr[X-CURSES]</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>mvgetch[X-CURSES]</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>slk_set[X-CURSES]</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>Function</td>
<td>Library</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------</td>
</tr>
<tr>
<td>addnstr</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>addstr</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>attr_get</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>attr_off</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>attr_on</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>attr_set</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>attroff</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>attron</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>attset</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>baudrate</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>beep</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>bkgd</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>bkgdset</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>border</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>box</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>can_change_color</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>cbreak</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>chgat</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>clear</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>clearok</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>clrtoeol</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>color_content</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>color_set</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>copywin</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>curs_set</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>def_prog_mode</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>def_shell_mode</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>def_curterm</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>delay_output</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>delch</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>deletemn</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>delscreen</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>delwin</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>derwin</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>dupdate</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>dupwin</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>echo</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>echochar</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>newpad</td>
<td>X-CURSES</td>
</tr>
<tr>
<td>Function</td>
<td>Function</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>endwin[X-CURSES]</td>
<td>newterm[X-CURSES]</td>
</tr>
<tr>
<td>erase[X-CURSES]</td>
<td>newwin[X-CURSES]</td>
</tr>
<tr>
<td>erasechar[X-CURSES]</td>
<td>nl[X-CURSES]</td>
</tr>
<tr>
<td>flash[X-CURSES]</td>
<td>nodelay[X-CURSES]</td>
</tr>
<tr>
<td>flushinp[X-CURSES]</td>
<td>noecho[X-CURSES]</td>
</tr>
<tr>
<td>getbkgd[X-CURSES]</td>
<td>nonl[X-CURSES]</td>
</tr>
<tr>
<td>getchnstr[X-CURSES]</td>
<td>noqiflush[X-CURSES]</td>
</tr>
<tr>
<td>getwin[X-CURSES]</td>
<td>noraw[X-CURSES]</td>
</tr>
<tr>
<td>idcok[X-CURSES]</td>
<td>pair_content[X-CURSES]</td>
</tr>
<tr>
<td>idlok[X-CURSES]</td>
<td>pechochar[X-CURSES]</td>
</tr>
<tr>
<td>immedok[X-CURSES]</td>
<td>prefresh[X-CURSES]</td>
</tr>
<tr>
<td>inch[X-CURSES]</td>
<td>printw[X-CURSES]</td>
</tr>
<tr>
<td>inchnstr[LSB]</td>
<td>putp[X-CURSES]</td>
</tr>
<tr>
<td>ini[nstr][LSB]</td>
<td>putwin[X-CURSES]</td>
</tr>
<tr>
<td>init_color[LSB]</td>
<td>refresh[X-CURSES]</td>
</tr>
<tr>
<td>init_pair[X-CURSES]</td>
<td>reset_prog_mode[X-</td>
</tr>
<tr>
<td></td>
<td>CURSES]</td>
</tr>
<tr>
<td>init_scr[X-CURSES]</td>
<td>reset_shell_mode[X-</td>
</tr>
<tr>
<td></td>
<td>CURSES]</td>
</tr>
<tr>
<td>innstr[X-CURSES]</td>
<td>resety[X-CURSES]</td>
</tr>
<tr>
<td>insch[X-CURSES]</td>
<td>restartterm[X-CURSES]</td>
</tr>
<tr>
<td>insdelln[X-CURSES]</td>
<td>ripoffline[LSB]</td>
</tr>
<tr>
<td>insertln[X-CURSES]</td>
<td>savetty[X-CURSES]</td>
</tr>
<tr>
<td>intrflush[X-CURSES]</td>
<td>scanw[LSB]</td>
</tr>
<tr>
<td>is_line_touched[X-</td>
<td>scr_dump[X-CURSES]</td>
</tr>
<tr>
<td></td>
<td>CURSES]</td>
</tr>
<tr>
<td>is_win_touched[X-</td>
<td>scr_init[X-CURSES]</td>
</tr>
<tr>
<td></td>
<td>CURSES]</td>
</tr>
<tr>
<td>isendwin[X-CURSES]</td>
<td>scr_restore[X-CURSES]</td>
</tr>
<tr>
<td>keyname[X-CURSES]</td>
<td>scr_set[X-CURSES]</td>
</tr>
<tr>
<td>keypad[X-CURSES]</td>
<td>scroll[X-CURSES]</td>
</tr>
<tr>
<td>killchar[X-CURSES]</td>
<td>set_curterm[X-CURSES]</td>
</tr>
<tr>
<td></td>
<td>set_term[X-CURSES]</td>
</tr>
<tr>
<td></td>
<td>wrefresh[X-CURSES]</td>
</tr>
</tbody>
</table>
### A.7 libncursesw

The behavior of the interfaces in this library is specified by the following Standards.  

**Libncursesw API** [Libncursesw]  
**This Specification** [LSB]  
**Libncursesw Placeholder** [ncursesw]  
**X/Open Curses, Issue 7** [X-CURSES]

#### Table A-10 libncursesw Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>add_wch[Libncursesw]</td>
<td>Add a character to the terminal window</td>
</tr>
<tr>
<td>mvaddnwstr[Libncursesw]</td>
<td>Move and add a wide character string</td>
</tr>
<tr>
<td>add_wchnstr[Libncursesw]</td>
<td>Add a wide character string to the terminal window</td>
</tr>
<tr>
<td>add_wchstr[Libncursesw]</td>
<td>Add a character string to the terminal window</td>
</tr>
<tr>
<td>mvaddwstr[Libncursesw]</td>
<td>Move and add a wide character string</td>
</tr>
<tr>
<td>addch[Libncursesw]</td>
<td>Add a character to the terminal window</td>
</tr>
<tr>
<td>mvchgat[Libncursesw]</td>
<td>Move and change attribute to the terminal window</td>
</tr>
<tr>
<td>addchnstr[Libncursesw]</td>
<td>Add a character string to the terminal window</td>
</tr>
<tr>
<td>mvdelch[Libncursesw]</td>
<td>Move and delete a character from the terminal window</td>
</tr>
<tr>
<td>addnstr[Libncursesw]</td>
<td>Add a character string to the terminal window</td>
</tr>
<tr>
<td>mvderwin[Libncursesw]</td>
<td>Move and delete a window from the terminal window</td>
</tr>
<tr>
<td>addnwsrrr[Libncursesw]</td>
<td>Add a wide character string to the terminal window</td>
</tr>
<tr>
<td>mvget_wch[Libncursesw]</td>
<td>Move and get a character from the terminal window</td>
</tr>
<tr>
<td>addstr[Libncursesw]</td>
<td>Add a string to the terminal window</td>
</tr>
<tr>
<td>mvget_wstr[Libncursesw]</td>
<td>Move and get a string from the terminal window</td>
</tr>
<tr>
<td>addwstr[Libncursesw]</td>
<td>Add a string to the terminal window</td>
</tr>
<tr>
<td>mvgetch[Libncursesw]</td>
<td>Move and get a character from the terminal window</td>
</tr>
<tr>
<td>addstr[Libncursesw]</td>
<td>Add a string to the terminal window</td>
</tr>
<tr>
<td>mvgetn_wstr[Libncursesw]</td>
<td>Move and get a wide string from the terminal window</td>
</tr>
<tr>
<td>assume_default_colors[Libncursesw]</td>
<td>Assume default colors</td>
</tr>
<tr>
<td>attr_get[Libncursesw]</td>
<td>Get an attribute from the terminal window</td>
</tr>
<tr>
<td>attr_off[Libncursesw]</td>
<td>Turn off an attribute in the terminal window</td>
</tr>
<tr>
<td>attr_on[Libncursesw]</td>
<td>Turn on an attribute in the terminal window</td>
</tr>
<tr>
<td>assume_default_colors[Libncursesw]</td>
<td>Assume default colors</td>
</tr>
<tr>
<td>attr_get[Libncursesw]</td>
<td>Get an attribute from the terminal window</td>
</tr>
<tr>
<td>attr_off[Libncursesw]</td>
<td>Turn off an attribute in the terminal window</td>
</tr>
<tr>
<td>attr_on[Libncursesw]</td>
<td>Turn on an attribute in the terminal window</td>
</tr>
<tr>
<td>Function</td>
<td>Library</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>attr_set</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>attroff</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>atron</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>attrset</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>baudrate</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>beep</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>bkgd</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>bkgdset</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>bkgrnd</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>border</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>border_set</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>box</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>box_set</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>can_change_color</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>cbreak</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>chgat</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>clear</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>clearok</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>clrtoeol</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>color_content</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>color_set</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>copywin</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>curs_set</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>curses_version</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>def_prog_mode</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>def_shell_mode</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>define_key</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>del_curterm</td>
<td>Libcursesw</td>
</tr>
<tr>
<td>delay_output</td>
<td>Libcursesw</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
<table>
<thead>
<tr>
<th>Function</th>
<th>Library</th>
<th>Function</th>
<th>Library</th>
</tr>
</thead>
<tbody>
<tr>
<td>delch</td>
<td>[Libcursesw]</td>
<td>mvwchgat</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>deleteln</td>
<td>[Libcursesw]</td>
<td>mvwdelch</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>delscreen</td>
<td>[Libcursesw]</td>
<td>mvwget_wch</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>delwin</td>
<td>[Libcursesw]</td>
<td>mvwget_wstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>derwin</td>
<td>[Libcursesw]</td>
<td>mvwgetch</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>doupdate</td>
<td>[Libcursesw]</td>
<td>mvwgetn_wstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>dupwin</td>
<td>[Libcursesw]</td>
<td>mvwgetnstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>echo</td>
<td>[Libcursesw]</td>
<td>mvwgetstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>echo_wchar</td>
<td></td>
<td>mvwhline</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>echochar</td>
<td>[Libcursesw]</td>
<td>mvwhline_set</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>endwin</td>
<td>[Libcursesw]</td>
<td>mvwin</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>erase</td>
<td>[Libcursesw]</td>
<td>mvwin_wch</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>erasechar</td>
<td>[Libcursesw]</td>
<td>mvwin_wchnstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>erasewchar</td>
<td>[Libcursesw]</td>
<td>mvwin_wchstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>filter</td>
<td>[Libcursesw]</td>
<td>mvwinch</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>flash</td>
<td>[Libcursesw]</td>
<td>mvwinchestr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>flushinp</td>
<td>[Libcursesw]</td>
<td>mvwinchstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>get_wch</td>
<td>[Libcursesw]</td>
<td>mvwinnstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>get_wstr</td>
<td>[Libcursesw]</td>
<td>mvwinnwstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>getbkgd</td>
<td>[Libcursesw]</td>
<td>mwwins_nwstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>getbkgnd</td>
<td></td>
<td>mwwins_wch</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>getcchar</td>
<td>[Libcursesw]</td>
<td>mwwins_wstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>getch</td>
<td>[Libcursesw]</td>
<td>mwwinsch</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>getmouse</td>
<td>[Libcursesw]</td>
<td>mwwinsnstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>getn_wstr</td>
<td>[Libcursesw]</td>
<td>mwwinstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>getnstr</td>
<td>[Libcursesw]</td>
<td>mwwinstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>getstr</td>
<td>[Libcursesw]</td>
<td>mvwinwstr</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>getwin</td>
<td>[Libcursesw]</td>
<td>mwwprintw</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>halfdelay</td>
<td>[Libcursesw]</td>
<td>mwwscanw</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>has_colors</td>
<td>[Libcursesw]</td>
<td>mvwvline</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>has_ic</td>
<td>[Libcursesw]</td>
<td>mvwvline_set</td>
<td>[Libcursesw]</td>
</tr>
<tr>
<td>Function</td>
<td>Library</td>
<td>Function</td>
<td>Library</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------------------------</td>
<td>-----------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>has_ii</td>
<td>Libncursesw</td>
<td>napms</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>has_key</td>
<td>Libncursesw</td>
<td>wcolor_set</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>has_mouse</td>
<td>LSB</td>
<td>newpad</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wdsyncup</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>hline</td>
<td>Libncursesw</td>
<td>newwin</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wdeleteln</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>hline_set</td>
<td>Libncursesw</td>
<td>nl</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wecho_wchar</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>has_key</td>
<td>Libncursesw</td>
<td>nocbreak</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>nodelay</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>idcok</td>
<td>Libncursesw</td>
<td>noecho</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wget_wch</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>in_wch</td>
<td>Libncursesw</td>
<td>nonl</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wget_str</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>hline</td>
<td>Libncursesw</td>
<td>pair_content</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>whline</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>init_color</td>
<td>Libncursesw</td>
<td>pecho_wchar</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>whline_set</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>init_pair</td>
<td>Libncursesw</td>
<td>pechochar</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>win_wch</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>initscr</td>
<td>Libncursesw</td>
<td>pncrntrefresh</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>win_wchnstr</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>inchnstr</td>
<td>Libncursesw</td>
<td>prefetch</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>win_wchstr</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>innwstr</td>
<td>Libncursesw</td>
<td>printw</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winch</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>ins_wch</td>
<td>Libncursesw</td>
<td>putpt</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winchnstr</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>ins_wstr</td>
<td>Libncursesw</td>
<td>putwin</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winchstr</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>instr</td>
<td>Libncursesw</td>
<td>qiflush</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winnstr</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>insdelln</td>
<td>Libncursesw</td>
<td>raw</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winnstrw</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>insertln</td>
<td>Libncursesw</td>
<td>redrawwin</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wins_nwstr</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>insnstr</td>
<td>Libncursesw</td>
<td>refresh</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wins_wch</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>insstr</td>
<td>Libncursesw</td>
<td>reset_prog_mode</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wins_wstr</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>instr</td>
<td>Libncursesw</td>
<td>reset_shell_mode</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winsch</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>intrflush</td>
<td>Libncursesw</td>
<td>resetty</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winsdelln</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>inwstr</td>
<td>Libncursesw</td>
<td>resizeterm</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winsertln</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>is_line_touched</td>
<td>Libncursesw</td>
<td>restarterterm</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winsnstr</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>is_win_touched</td>
<td>Libncursesw</td>
<td>ripoff_line</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winsyll</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>isendwin</td>
<td>Libncursesw</td>
<td>savetty</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winsrr</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>key_name</td>
<td>Libncursesw</td>
<td>scanw</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>winwstr</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>keyboard</td>
<td>Libncursesw</td>
<td>scr_dump</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wmouse_trafo</td>
<td>Libncursesw</td>
</tr>
<tr>
<td>keyname</td>
<td>Libncursesw</td>
<td>scr_init</td>
<td>Libncursesw</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wmove</td>
<td>Libncursesw</td>
</tr>
</tbody>
</table>
Annex A Alphabetical Listing of Interfaces by Library

<table>
<thead>
<tr>
<th>Library</th>
<th>Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSB Core - Generic 5.0</td>
<td>keyok</td>
</tr>
<tr>
<td></td>
<td>Libncursesw</td>
</tr>
</tbody>
</table>

A.8 libpam

The behavior of the interfaces in this library is specified by the following Standards.

This Specification [LSB]
PAM [PAM]

Table A-12 libpam Function Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>LIBPAM_1.0</th>
<th>LIBPAM_1.0</th>
<th>LIBPAM_1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>pam_acct_mgmt</td>
<td>[LSB]</td>
<td>[PAM]</td>
<td>[LSB]</td>
</tr>
<tr>
<td>pam_authenticate</td>
<td>[LSB]</td>
<td>[PAM]</td>
<td>[LSB]</td>
</tr>
<tr>
<td>pam_chauthtok</td>
<td>[LSB]</td>
<td>[PAM]</td>
<td>[LSB]</td>
</tr>
<tr>
<td>pam_close_session</td>
<td>[LSB]</td>
<td>[PAM]</td>
<td>[LSB]</td>
</tr>
<tr>
<td>pam_get_data</td>
<td>[LIBPAM_1.0]</td>
<td>[PAM]</td>
<td>[PAM]</td>
</tr>
<tr>
<td>pam_get_item</td>
<td>[LIBPAM_1.0]</td>
<td>[PAM]</td>
<td>[PAM]</td>
</tr>
<tr>
<td>pam_get_user</td>
<td>[LIBPAM_1.0]</td>
<td>[PAM]</td>
<td>[PAM]</td>
</tr>
<tr>
<td>pam_getenv</td>
<td>[LIBPAM_1.0]</td>
<td>[PAM]</td>
<td>[PAM]</td>
</tr>
<tr>
<td>pam_putenv</td>
<td>[LIBPAM_1.0]</td>
<td>[PAM]</td>
<td>[PAM]</td>
</tr>
<tr>
<td>pam_set_data</td>
<td>[LIBPAM_1.0]</td>
<td>[PAM]</td>
<td>[PAM]</td>
</tr>
<tr>
<td>pam_set_item</td>
<td>[LIBPAM_1.0]</td>
<td>[PAM]</td>
<td>[PAM]</td>
</tr>
<tr>
<td>pam_setcred</td>
<td>[LIBPAM_1.0]</td>
<td>[PAM]</td>
<td>[PAM]</td>
</tr>
</tbody>
</table>

Table A-11 libncursesw Data Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>LIBNCURSESW</th>
<th>NCURSESW</th>
<th>NCURSESW</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLORS</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>COLOR_PAIRS</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>COLS</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>LINES</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>acs_map</td>
<td>[LSB]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>cur_term</td>
<td>[LSB]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>stdscr</td>
<td>[LSB]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>ttytype</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>slk_attr</td>
<td>[X-CURSES]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>slk_attr_off</td>
<td>[X-CURSES]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>slk_attr_on</td>
<td>[X-CURSES]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>slk_attr_set</td>
<td>[Libncursesw]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>slk_attroff</td>
<td>[Libncursesw]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>slk_atrtron</td>
<td>[Libncursesw]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>slk_attrset</td>
<td>[Libncursesw]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>slk_color</td>
<td>[Libncursesw]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
<tr>
<td>newscr</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
<td>[ncursesw]</td>
</tr>
</tbody>
</table>
A.9 libpthread

The behavior of the interfaces in this library is specified by the following Standards.

- Large File Support [LFS]
- This Specification [LSB]

### Table A-13 libpthread Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Standard</th>
<th>Function</th>
<th>Standard</th>
<th>Function</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>__errno_location</td>
<td>LSB</td>
<td>pthread_cond_destroy</td>
<td>SUSv4</td>
<td>pthread_rwlock_tryrdlock</td>
<td>SUSv4</td>
</tr>
<tr>
<td>__h_errno_location</td>
<td>LSB</td>
<td>pthread_cond_init</td>
<td>SUSv4</td>
<td>pthread_rwlock_trywlock</td>
<td>SUSv4</td>
</tr>
<tr>
<td>__libc_current_sigrtmax</td>
<td>LSB</td>
<td>pthread_cond_signal</td>
<td>SUSv4</td>
<td>pthread_rwlock_unlock</td>
<td>SUSv4</td>
</tr>
<tr>
<td>__libc_current_sigrtmin</td>
<td>LSB</td>
<td>pthread_cond_timedwait</td>
<td>SUSv4</td>
<td>pthread_rwlock_wrlk</td>
<td>SUSv4</td>
</tr>
<tr>
<td>pthread_cleanup_pop</td>
<td>LSB</td>
<td>pthread_cond_timedwait</td>
<td>SUSv4</td>
<td>pthread_rwlockattr_destroy</td>
<td>SUSv4</td>
</tr>
<tr>
<td>pthread_cleanup_push</td>
<td>LSB</td>
<td>pthread_cond_timedwait</td>
<td>SUSv4</td>
<td>pthread_rwlockattr_getkey_np</td>
<td>LSB</td>
</tr>
<tr>
<td>accept</td>
<td>SUSv4</td>
<td>pthread_condattr_destroy</td>
<td>SUSv4</td>
<td>pthread_rwlockattr_getpset</td>
<td>SUSv4</td>
</tr>
<tr>
<td>close</td>
<td>SUSv4</td>
<td>pthread_condattr_getpset</td>
<td>SUSv4</td>
<td>pthread_rwlockattr_getpset</td>
<td>SUSv4</td>
</tr>
<tr>
<td>connect</td>
<td>SUSv4</td>
<td>pthread_condattr_getpset</td>
<td>SUSv4</td>
<td>pthread_rwlockattr_getpset</td>
<td>SUSv4</td>
</tr>
<tr>
<td>flockfile</td>
<td>SUSv4</td>
<td>pthread_condattr_getpset</td>
<td>SUSv4</td>
<td>pthread_rwlockattr_getpset</td>
<td>SUSv4</td>
</tr>
<tr>
<td>fork</td>
<td>SUSv4</td>
<td>pthread_create</td>
<td>SUSv4</td>
<td>pthread_setcancelstate</td>
<td>SUSv4</td>
</tr>
<tr>
<td>fsync</td>
<td>SUSv4</td>
<td>pthread_detach</td>
<td>SUSv4</td>
<td>pthread_setcanceltype</td>
<td>SUSv4</td>
</tr>
<tr>
<td>ftrylockfile</td>
<td>SUSv4</td>
<td>pthread_equal</td>
<td>SUSv4</td>
<td>pthread_setconcurrency</td>
<td>SUSv4</td>
</tr>
<tr>
<td>flockfile</td>
<td>SUSv4</td>
<td>pthread_equal</td>
<td>SUSv4</td>
<td>pthread_setconcurrency</td>
<td>SUSv4</td>
</tr>
<tr>
<td>funlockfile</td>
<td>SUSv4</td>
<td>pthread_exit</td>
<td>SUSv4</td>
<td>pthread_setschedparam</td>
<td>SUSv4</td>
</tr>
<tr>
<td>lseek</td>
<td>SUSv4</td>
<td>pthread_getattr_np</td>
<td>LSB</td>
<td>pthread_setschedprio</td>
<td>GLIBC_2.3.4</td>
</tr>
<tr>
<td>lseek64</td>
<td>LFS</td>
<td>pthread_getattr_np</td>
<td>LSB</td>
<td>pthread_setschedprio</td>
<td>GLIBC_2.3.4</td>
</tr>
<tr>
<td>msync</td>
<td>SUSv4</td>
<td>pthread_getsockopt</td>
<td>SUSv4</td>
<td>pthread_setspecific</td>
<td>SUSv4</td>
</tr>
<tr>
<td>msync</td>
<td>SUSv4</td>
<td>pthread_getsockopt</td>
<td>SUSv4</td>
<td>pthread_setspecific</td>
<td>SUSv4</td>
</tr>
<tr>
<td>Function</td>
<td>SUSv4</td>
<td>Sv4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------</td>
<td>-----</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pread64[LSB]</td>
<td>pthread_mutex_consistent(GLIBC_2.12)[SUSv4]</td>
<td>pwrite[SUSv4]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### A.10 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-14 librt Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Standard</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aio_cancel</td>
<td>SUSv4</td>
<td>mq_send(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>aio_cancel64</td>
<td>LFS</td>
<td>clock_gettime(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>aio_error</td>
<td>SUSv4</td>
<td>clock_gettime(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>aio_error64</td>
<td>LFS</td>
<td>clock_gettime(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>aio_fsync</td>
<td>SUSv4</td>
<td>clock_getres(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>aio_fsync64</td>
<td>LFS</td>
<td>clock_getres(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>aio_read</td>
<td>SUSv4</td>
<td>lio_listio(GLIBC_2.4)</td>
</tr>
<tr>
<td>aio_read64</td>
<td>LFS</td>
<td>lio_listio64(GLIBC_2.4)</td>
</tr>
<tr>
<td>aio_write</td>
<td>SUSv4</td>
<td>mq_setattr(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>aio_write64</td>
<td>LFS</td>
<td>mq_setattr(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>clock_getcpuclockid</td>
<td>SUSv4</td>
<td>mq_timedreceive(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>clock_getres</td>
<td>SUSv4</td>
<td>mq_timedsend(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>clock_gettime</td>
<td>SUSv4</td>
<td>mq_unlink(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>clock_gettime</td>
<td>SUSv4</td>
<td>clock_gettime(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>clock_nanosleep</td>
<td>SUSv4</td>
<td>clock_gettime(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>clock_settime</td>
<td>SUSv4</td>
<td>clock_gettime(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>shm_open</td>
<td>SUSv4</td>
<td>shm_open(GLIBC_2.3.4)</td>
</tr>
<tr>
<td>shm_unlink</td>
<td>SUSv4</td>
<td>shm_unlink(GLIBC_2.3.4)</td>
</tr>
</tbody>
</table>

© 2015 Linux Foundation
Annex A Alphabetical Listing of Interfaces by Library

A.11 libutil

The behavior of the interfaces in this library is specified by the following Standards.

This Specification [LSB]

Table A-15 libutil Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>LSB</th>
<th>Function</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>forkpty</td>
<td>[LSB]</td>
<td>login_tty</td>
<td>[LSB]</td>
</tr>
<tr>
<td>login</td>
<td>[LSB]</td>
<td>logout</td>
<td>[LSB]</td>
</tr>
<tr>
<td>logwtmp</td>
<td>[LSB]</td>
<td>openpty</td>
<td>[LSB]</td>
</tr>
</tbody>
</table>

A.12 libz

The behavior of the interfaces in this library is specified by the following Standards.

This Specification [LSB]

Table A-16 libz Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>LSB</th>
<th>Function</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>adler32</td>
<td>[LSB]</td>
<td>gzclose</td>
<td>[LSB]</td>
</tr>
<tr>
<td>compress</td>
<td>[LSB]</td>
<td>gzdopen</td>
<td>[LSB]</td>
</tr>
<tr>
<td>compress2</td>
<td>[LSB]</td>
<td>gzeof</td>
<td>[LSB]</td>
</tr>
<tr>
<td>compressBound</td>
<td>ZLIB_1.2.0</td>
<td>gerror</td>
<td>[LSB]</td>
</tr>
<tr>
<td>crc32</td>
<td>[LSB]</td>
<td>gzflush</td>
<td>[LSB]</td>
</tr>
<tr>
<td>deflate</td>
<td>[LSB]</td>
<td>gzgetc</td>
<td>[LSB]</td>
</tr>
<tr>
<td>deflateBound</td>
<td>ZLIB_1.2.0</td>
<td>gzgets</td>
<td>[LSB]</td>
</tr>
<tr>
<td>deflateCopy</td>
<td>[LSB]</td>
<td>gzopen</td>
<td>[LSB]</td>
</tr>
<tr>
<td>deflateEnd</td>
<td>[LSB]</td>
<td>gzprintf</td>
<td>[LSB]</td>
</tr>
<tr>
<td>deflateInit</td>
<td>[LSB]</td>
<td>gzputc</td>
<td>[LSB]</td>
</tr>
<tr>
<td>deflateInit2</td>
<td>[LSB]</td>
<td>gzputs</td>
<td>[LSB]</td>
</tr>
<tr>
<td>deflateParams</td>
<td>[LSB]</td>
<td>gzread</td>
<td>[LSB]</td>
</tr>
<tr>
<td>deflatePrime</td>
<td>ZLIB_1.2.0.8</td>
<td>gzrewind</td>
<td>[LSB]</td>
</tr>
<tr>
<td>deflateReset</td>
<td>[LSB]</td>
<td>gzseek</td>
<td>[LSB]</td>
</tr>
<tr>
<td>deflateSetDictionary</td>
<td>[LSB]</td>
<td>gsetparams</td>
<td>[LSB]</td>
</tr>
<tr>
<td>get_crc_table</td>
<td>[LSB]</td>
<td>gztell</td>
<td>[LSB]</td>
</tr>
<tr>
<td>gzclearerr</td>
<td>ZLIB_1.2.0.2</td>
<td>gzwrite</td>
<td>[LSB]</td>
</tr>
</tbody>
</table>
The behavior of the interfaces in this library is specified by the following Standards.

**NSPR Reference** [NSPR]

### Table A-17 libnspr4 Function Interfaces

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Function</th>
<th>Description</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR_AtomicIncrement</td>
<td>[NSPR]</td>
<td>PR_GetThreadPrivate</td>
<td>[NSPR]</td>
<td>PR_OpenTCP</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_AtomicSet</td>
<td>[NSPR]</td>
<td>PR_GetThreadScope</td>
<td>[NSPR]</td>
<td>PR_OpenUDP</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_Cleanup</td>
<td>[NSPR]</td>
<td>PR_Implode</td>
<td>[NSPR]</td>
<td>PR_Realloc</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_Close</td>
<td>[NSPR]</td>
<td>PR_Init</td>
<td>[NSPR]</td>
<td>PR_Pipe</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_Connect</td>
<td>[NSPR]</td>
<td>PR_Initiate</td>
<td>[NSPR]</td>
<td>PR_Poll</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_ConnectContinue</td>
<td>[NSPR]</td>
<td>PR_InitiateNetAddr</td>
<td>[NSPR]</td>
<td>PR_PopIOLayer</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_ConvertIPv4AddrToIPv6</td>
<td>[NSPR]</td>
<td>PR_Initial</td>
<td>[NSPR]</td>
<td>PR_realloc</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_CreateIOLayerStub</td>
<td>[NSPR]</td>
<td>PR_Interrupt</td>
<td>[NSPR]</td>
<td>PR_Recv</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_CreatePipe</td>
<td>[NSPR]</td>
<td>PR_Interval</td>
<td>[NSPR]</td>
<td>PR_RecvFrom</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_DestroyCondVar</td>
<td>[NSPR]</td>
<td>PR_IntervalToMicroseconds</td>
<td>[NSPR]</td>
<td>PR_Send</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_DestroyLock</td>
<td>[NSPR]</td>
<td>PR_IntervalToMilliseconds</td>
<td>[NSPR]</td>
<td>PR_SendTo</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_EnumerateAddrInfo</td>
<td>[NSPR]</td>
<td>PR_IntervalToSeconds</td>
<td>[NSPR]</td>
<td>PR_SetError</td>
<td>[NSPR]</td>
</tr>
<tr>
<td>PR_GMTParameters</td>
<td>[NSPR]</td>
<td>PR_MicrosecondsToInt</td>
<td>[NSPR]</td>
<td>PR_Shutdown</td>
<td>[NSPR]</td>
</tr>
</tbody>
</table>
### A.14 libnss3

The behavior of the interfaces in this library is specified by the following Standards.

 Mozilla's NSS SSL Reference [NSS SSL]

#### Table A-18 libnss3 Function Interfaces

<table>
<thead>
<tr>
<th>Function Call</th>
<th>NSS SSL Reference</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERT_CHECKCERTVALIDTIMES(NSS_3.2)</td>
<td>NSS SSL</td>
<td>CERT_CHECKCERTVALIDTIMES(NSS_3.2)</td>
</tr>
<tr>
<td>CERT_VERIFYCERTNOW(NSS_3.2)</td>
<td>NSS SSL</td>
<td>CERT_VERIFYCERTNOW(NSS_3.2)</td>
</tr>
<tr>
<td>PK11_GETSLOTNAME(NSS_3.2)</td>
<td>NSS SSL</td>
<td>PK11_GETSLOTNAME(NSS_3.2)</td>
</tr>
<tr>
<td>CERT_DESTROYCERTIFICATE(NSS_3.2)</td>
<td>NSS SSL</td>
<td>CERT_DESTROYCERTIFICATE(NSS_3.2)</td>
</tr>
<tr>
<td>NSS_INIT(NSS_3.2)</td>
<td>NSS SSL</td>
<td>NSS_INIT(NSS_3.2)</td>
</tr>
<tr>
<td>PK11_GETTOKENNAME(NSS_3.2)</td>
<td>NSS SSL</td>
<td>PK11_GETTOKENNAME(NSS_3.2)</td>
</tr>
<tr>
<td>CERT_DUPCERTIFICATE(NSS_3.2)</td>
<td>NSS SSL</td>
<td>CERT_DUPCERTIFICATE(NSS_3.2)</td>
</tr>
<tr>
<td>NSS_INITREADWRITE(NSS_3.2)</td>
<td>NSS SSL</td>
<td>NSS_INITREADWRITE(NSS_3.2)</td>
</tr>
<tr>
<td>PK11_ISHW(NSS_3.2)</td>
<td>NSS SSL</td>
<td>PK11_ISHW(NSS_3.2)</td>
</tr>
<tr>
<td>CERT_FREE Nicknames(NSS_3.2)</td>
<td>NSS SSL</td>
<td>CERT_FREE Nicknames(NSS_3.2)</td>
</tr>
<tr>
<td>NSS_NODB_INIT(NSS_3.2)</td>
<td>NSS SSL</td>
<td>NSS_NODB_INIT(NSS_3.2)</td>
</tr>
<tr>
<td>PK11_ISPRESENT(NSS_3.2)</td>
<td>NSS SSL</td>
<td>PK11_ISPRESENT(NSS_3.2)</td>
</tr>
<tr>
<td>CERT_GETCERTnickname(NSS_3.2)</td>
<td>NSS SSL</td>
<td>CERT_GETCERTnickname(NSS_3.2)</td>
</tr>
<tr>
<td>NSS_SHUTDOWN(NSS_3.2)</td>
<td>NSS SSL</td>
<td>NSS_SHUTDOWN(NSS_3.2)</td>
</tr>
<tr>
<td>PK11_ISREADONLY(NSS_3.2)</td>
<td>NSS SSL</td>
<td>PK11_ISREADONLY(NSS_3.2)</td>
</tr>
<tr>
<td>CERT_GetDefaultCertDB(NSS_3.2)</td>
<td>NSS SSL</td>
<td>CERT_GetDefaultCertDB(NSS_3.2)</td>
</tr>
<tr>
<td>PK11_FINDCERTFROMNickname(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>PK11_FINDCERTFROMNickname(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>PK11_SETPASSWORDFUNC(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>PK11_SETPASSWORDFUNC(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>CERT_VERIFYCERTNAME(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>CERT_VERIFYCERTNAME(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>PK11_FINDKEYBYAnyCertificate(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>PK11_FINDKEYBYAnyCertificate(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>SECKEY_DESTROYPRIVATEKey(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>SECKEY_DESTROYPRIVATEKey(NSS_3.2)[NSS SSL]</td>
</tr>
</tbody>
</table>

### A.15 libssl3

The behavior of the interfaces in this library is specified by the following Standards.

 Mozilla's NSS SSL Reference [NSS SSL]

#### Table A-19 libssl3 Function Interfaces

<table>
<thead>
<tr>
<th>Function Call</th>
<th>NSS SSL Reference</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSS_CMPCERTCHAINWCANNAMES(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>SSL_CMPCERTCHAINWCANNAMES(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>SSL_CONFIGSERVERSIDLICENSE(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>SSL_CONFIGSERVERSIDLICENSE(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>SSL_OPTIONSET(NSS_3.2)</td>
<td>NSS SSL</td>
<td>SSL_OPTIONSET(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_CONFIGSECURESERVER(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>SSL_CONFIGSECURESERVER(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>SSL_OPTIONSETDEFAULT(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>SSL_OPTIONSETDEFAULT(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>SSL_CONFIGSERVERSESSIONIDCACHE(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>SSL_CONFIGSERVERSESSIONIDCACHE(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>SSL_PEERCERTIFICATE(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>SSL_PEERCERTIFICATE(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>SSL_AuthCertificate(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>SSL_AuthCertificate(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>SSL_DATA_PENDING(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>SSL_DATA_PENDING(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>SSL_REHANDSHAKE(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>SSL_REHANDSHAKE(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>SSL_RESETHANDSHAKE(NSS_3.2)[NSS SSL]</td>
<td>NSS SSL</td>
<td>SSL_RESETHANDSHAKE(NSS_3.2)[NSS SSL]</td>
</tr>
<tr>
<td>Function</td>
<td>NSS SSL Version</td>
<td>Function</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>-----------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>SSL_BadCertHook(NSS_3.2)</td>
<td>[NSS SSL]</td>
<td>SSL_GetClientAuthDataHook(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_CIPHER_POLICY_GET(NSS_3.2)</td>
<td>[NSS SSL]</td>
<td>SSL_GetSessionID(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_CIPHER_POLICY_SET(NSS_3.2)</td>
<td>[NSS SSL]</td>
<td>SSL_HandshakeCallback(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_CIPHER_PREF_GET(NSS_3.2)</td>
<td>[NSS SSL]</td>
<td>SSL_IMPORTFD(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_CIPHER_PREF_GET_DEFAULT(NSS_3.2)</td>
<td>[NSS SSL]</td>
<td>SSL_INHERIT_MPSERVERSID_CACHE(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_CIPHER_PREF_SET(NSS_3.2)</td>
<td>[NSS SSL]</td>
<td>SSL_INVALIDATE_SESSION(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_CIPHER_PREF_SET_DEFAULT(NSS_3.2)</td>
<td>[NSS SSL]</td>
<td>SSL_OPTION_GET(NSS_3.2)</td>
</tr>
<tr>
<td>SSL_CLEAR_SESSIONCACHE(NSS_3.2)</td>
<td>[NSS SSL]</td>
<td>SSL_OPTION_GET_DEFAULT(NSS_3.2)</td>
</tr>
</tbody>
</table>
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. Secondarily, 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"
BS Core - Generic 5.0  Annex B GNU Free Documentation

License (Informative)

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.