

# **Linux Standard Base C++ Specification 3.1**

## **Linux Standard Base C++ Specification 3.1**

Copyright © 2004, 2005, 2006 Free Standards Group

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

Portions of the text are copyrighted by the following parties:

- The Regents of the University of California
- Free Software Foundation
- Ian F. Darwin
- Paul Vixie
- BSDI (now Wind River)
- Andrew G Morgan
- Jean-loup Gailly and Mark Adler
- Massachusetts Institute of Technology

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

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

UNIX is a registered trademark of The Open Group.

LSB is a trademark of the Free Standards Group in the United States and other countries.

AMD is a trademark of Advanced Micro Devices, Inc.

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

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

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

OpenGL is a registered trademark of Silicon Graphics, Inc.

# Contents

<b>I Introductory Elements .....</b>	<b>1</b>
1 Scope.....	1
1.1 General.....	1
1.2 Module Specific Scope.....	1
2 Normative References.....	2
3 Requirements .....	3
3.1 Relevant Libraries .....	3
3.2 LSB Implementation Conformance .....	3
3.3 LSB Application Conformance.....	4
4 Definitions .....	5
5 Terminology .....	6
6 Documentation Conventions .....	8
<b>II Low Level System Information .....</b>	<b>9</b>
7 C++ Class Representations.....	10
7.1 C++ Data Representation.....	10
8 Symbol Mapping .....	13
8.1 Symbol Mapping.....	13
<b>III Base Libraries .....</b>	<b>14</b>
9 Libraries .....	15
9.1 Interfaces for libstdcxx.....	15
9.2 Interface Definitions for libstdcxx.....	235
<b>A GNU Free Documentation License (Informative).....</b>	<b>236</b>
A.1 PREAMBLE .....	236
A.2 APPLICABILITY AND DEFINITIONS .....	236
A.3 VERBATIM COPYING .....	237
A.4 COPYING IN QUANTITY .....	237
A.5 MODIFICATIONS.....	238
A.6 COMBINING DOCUMENTS .....	239
A.7 COLLECTIONS OF DOCUMENTS .....	240
A.8 AGGREGATION WITH INDEPENDENT WORKS .....	240
A.9 TRANSLATION.....	240
A.10 TERMINATION.....	240
A.11 FUTURE REVISIONS OF THIS LICENSE .....	241
A.12 How to use this License for your documents .....	241

## List of Figures

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

## Foreword

This is version 3.1 of the Linux Standard Base C++ Specification. This specification is part of a family of specifications under the general title "Linux Standard Base". Developers of applications or implementations interested in using the LSB trademark should see the Free Standards Group Certification Policy for details.

## Introduction

The LSB defines a binary interface for application programs that are compiled and packaged for LSB-conforming implementations on many different hardware architectures. Since a binary specification shall include information specific to the computer processor architecture for which it is intended, it is not possible for a single document to specify the interface for all possible LSB-conforming implementations. Therefore, the LSB is a family of specifications, rather than a single one.

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:

- The first number ( $x$ ) is the major version number. All versions with the same major version number should share binary compatibility. Any addition or deletion of a new library results in a new version number. Interfaces marked as `deprecated` may be removed from the specification at a major version change.
- The second number ( $y$ ) is the minor version number. Individual interfaces may be added if all certified implementations already had that (previously undocumented) interface. Interfaces may be marked as `deprecated` at a minor version change. Other minor changes may be permitted at the discretion of the LSB workgroup.
- 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.

# 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 specification ("LSB-generic" or "generic LSB"), ISO/IEC 23360 Part 1, describing those parts of the interface that remain constant across all implementations of the LSB, and an architecture-specific part ("LSB-arch" or "archLSB") describing the parts of the interface that vary by processor architecture. Together, the LSB-generic and the relevant architecture-specific part of ISO/IEC 23360 for a single hardware architecture provide a complete interface specification for compiled application programs on systems that share a common hardware architecture.

ISO/IEC 23360 Part 1, the LSB-generic document, should be used in conjunction with an architecture-specific part. Whenever a section of the LSB-generic specification is supplemented by architecture-specific information, the LSB-generic document includes a reference to the architecture part. Architecture-specific parts of ISO/IEC 23360 may also contain additional information that is not referenced in the LSB-generic document.

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 C++ module of the Linux Standards Base (LSB). This module supplements the core interfaces by providing system interfaces, libraries, and a runtime environment for applications built using the C++ programming language. These interfaces provide low-level support for the core constructs of the language, and implement the standard base C++ libraries.

Interfaces described in this module are presented in terms of C++; the binary interfaces will use encoded or mangled versions of the names.

## 2 Normative References

The specifications listed below are referenced in whole or in part by this module of the Linux Standard Base. In this specification, where only a particular section of one of these references is identified, then the normative reference is to that section alone, and the rest of the referenced document is informative.

**Table 2-1 Normative References**

Name	Title	URL
ISO/IEC 23360 Part 1	ISO/IEC 23360:2005 Linux Standard Base - Part 1 Generic Specification	<a href="http://www.linuxbase.org/spec/">http://www.linuxbase.org/spec/</a>
ISO C (1999)	ISO/IEC 9899: 1999, Programming Languages --C	
ISO POSIX (2003)	ISO/IEC 9945-1:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 1: Base Definitions  ISO/IEC 9945-2:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 2: System Interfaces  ISO/IEC 9945-3:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 3: Shell and Utilities  ISO/IEC 9945-4:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 4: Rationale  Including Technical Cor. 1: 2004	<a href="http://www.unix.org/version3/">http://www.unix.org/version3/</a>
ISO/IEC 14882: 2003 C++ Language	ISO/IEC 14882: 2003 Programming languages --C++	
Itanium™ C++ ABI	Itanium™ C++ ABI (Revision 1.83)	<a href="http://refspecs.freestdards.org/cxxabi-1.83.html">http://refspecs.freestdards.org/cxxabi-1.83.html</a>

## 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.

Table 3-1 Standard Library Names

Library	Runtime Name
libstdcxx	libstdc++.so.6

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

### 3.2 LSB Implementation Conformance

An implementation shall satisfy the following requirements:

- The implementation shall implement fully the architecture described in the hardware manual for the target processor architecture.
- The implementation shall be capable of executing compiled applications having the format and using the system interfaces described in this document.
- The implementation shall provide libraries containing the interfaces specified by this document, 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 document.
- The map of virtual memory provided by the implementation shall conform to the requirements of this document.
- 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 document.
- 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 document in the format defined here and in other referenced documents. All commands and utilities shall behave as required by this document. The implementation shall also provide all mandatory components of an application's runtime environment that are included or referenced in this document.
- 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

An application shall satisfy the following requirements:

- Its executable files are either shell scripts or object files in the format defined for the Object File Format system interface.
- Its object files participate in dynamic linking as defined in the Program Loading and Linking System interface.
- It employs only the instructions, traps, and other low-level facilities defined in the Low-Level System interface as being for use by applications.
- If it requires any optional interface defined in this document in order to be installed or to execute successfully, the requirement for that optional interface is stated in the application's documentation.
- It does not use any interface or data format that is not required to be provided by a conforming implementation, unless:
  - If such an interface or data format is supplied by another application through direct invocation of that application during execution, that application is in turn an LSB conforming application.
  - The use of that interface or data format, as well as its source, is identified in the documentation of the application.
- It shall not use any values for a named interface that are reserved for vendor extensions.

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

## 4 Definitions

For the purposes of this document, the following definitions, as specified in the *ISO/IEC Directives, Part 2, 2001, 4th Edition*, apply:

can

be able to; there is a possibility of; it is possible to

cannot

be unable to; there is no possibility of; it is not possible to

may

is permitted; is allowed; is permissible

need not

it is not required that; no...is required

shall

is to; is required to; it is required that; has to; only...is permitted; it is necessary

shall not

is not allowed [permitted] [acceptable] [permissible]; is required to be not; is required that...be not; is not to be

should

it is recommended that; ought to

should not

it is not recommended that; ought not to

## 5 Terminology

For the purposes of this document, the following terms apply:

### archLSB

The architectural part of the LSB Specification which describes the specific parts of the interface that are platform specific. The archLSB is complementary to the gLSB.

### Binary Standard

The total set of interfaces that are available to be used in the compiled binary code of a conforming application.

### gLSB

The common part of the LSB Specification that describes those parts of the interface that remain constant across all hardware implementations of the LSB.

### 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

The set of interfaces that are available to be used in the source code of a conforming application.

### 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.

Other terms and definitions used in this document shall have the same meaning as defined in Chapter 3 of the Base Definitions volume of ISO POSIX (2003).

## 6 Documentation Conventions

Throughout this document, the following typographic conventions are used:

`function()`

the name of a function

**command**

the name of a command or utility

CONSTANT

a constant value

*parameter*

a parameter

variable

a variable

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

name

the name of the interface

(*symver*)

An optional symbol version identifier, if required.

[*refno*]

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

For example,

<code>forkpty(GLIBC_2.0) [SUSv3]</code>
---

refers to the interface named `forkpty()` with symbol version `GLIBC_2.0` that is defined in the `SUSv3` reference.

**Note:** Symbol versions are defined in the architecture specific parts of ISO/IEC 23360 only.



## **II Low Level System Information**

## 7 C++ Class Representations

### 7.1 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.

#### 7.1.1 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 7-1 describes these Class Components at a high level.

Table 7-1 Class Components

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

##### 7.1.1.1 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 7-1 and Category 2 (Non-virtual bases only) is further described in Figure 7-2. LSB conforming systems shall support these categories.

```
struct {  
    ptrdiff_t      baseobject;  
    const char    *typeinfo;
```

```

        fptr          virtfuncs[0];
    };

```

**Figure 7-1 Category 1 Virtual Table**

```

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

```

**Figure 7-2 Category 2 Virtual Table**

### 7.1.1.2 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 7-3 Run-Time Type Information Prefix**

```

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

```

**Figure 7-4 Run-Time Type Information For Classes with no base class**

```

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

```

**Figure 7-5 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 7-6 Run-Time Type Information for classes with multiple inheritance**

```

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

```

```
};
```

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

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

**Figure 7-8 Run-Time Type Information for pointer to member types**

## **8 Symbol Mapping**

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

### **8.1 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.

#### **8.1.1 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

## 9 Libraries

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

### 9.1 Interfaces for libstdc++

Table 9-1 defines the library name and shared object name for the libstdc++ library

**Table 9-1 libstdc++ Definition**

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

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] Itanium™ C++ ABI  
[ISOCXX] ISO/IEC 14882: 2003 C++ Language  
[LSB] ISO/IEC 23360 Part 1

#### 9.1.1 C++ Runtime Support

##### 9.1.1.1 Interfaces for C++ Runtime Support

An LSB conforming implementation shall provide the generic methods for C++ Runtime Support specified in Table 9-2, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-2 libstdc++ - C++ Runtime Support Function Interfaces**

<code>__gnu_cxx::__atomic_add(int volatile*, int)</code> (GLIBCXX_3.4) [CXXABI]
<code>__gnu_cxx::__exchange_and_add(int volatile*, int)</code> (GLIBCXX_3.4) [CXXABI]
<code>__gnu_cxx::__verbose_terminate_handler()</code> (CXXABI_1.3) [CXXABI]
<code>unexpected()</code> (GLIBCXX_3.4) [ISOCXX]
<code>set_terminate(void (*)())</code> (GLIBCXX_3.4) [ISOCXX]
<code>set_unexpected(void (*)())</code> (GLIBCXX_3.4) [ISOCXX]
<code>set_new_handler(void (*)())</code> (GLIBCXX_3.4) [ISOCXX]
<code>__throw_bad_cast()</code> (GLIBCXX_3.4) [ISOCXX]
<code>__throw_bad_alloc()</code> (GLIBCXX_3.4) [ISOCXX]
<code>__throw_bad_typeid()</code> (GLIBCXX_3.4) [ISOCXX]
<code>uncaught_exception()</code> (GLIBCXX_3.4) [ISOCXX]
<code>__throw_ios_failure(char const*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__throw_logic_error(char const*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__throw_range_error(char const*)</code> (GLIBCXX_3.4) [ISOCXX]

<code>__throw_domain_error(char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__throw_length_error(char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__throw_out_of_range(char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__throw_bad_exception()(GLIBCXX_3.4) [ISOCXX]</code>
<code>__throw_runtime_error(char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__throw_overflow_error(char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__throw_underflow_error(char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__throw_invalid_argument(char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>terminate()(GLIBCXX_3.4) [ISOCXX]</code>
<code>operator delete[](void*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>operator delete[](void*, nothrow_t const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>operator delete(void*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>operator delete(void*, nothrow_t const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__cxa_allocate_exception(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_bad_cast(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_bad_typeid(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_begin_catch(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_call_unexpected(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_current_exception_type(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_demangle(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_end_catch(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_free_exception(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_get_globals(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_get_globals_fast(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_guard_abort(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_guard_acquire(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_guard_release(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_pure_virtual(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_rethrow(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_throw(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_vec_ctor(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_vec_cleanup(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_vec_ctor(CXXABI_1.3) [CXXABI]</code>
<code>__cxa_vec_delete(CXXABI_1.3) [CXXABI]</code>



<code>__cxa_vec_delete2(CXXABI_1.3)</code> [CXXABI]
<code>__cxa_vec_delete3(CXXABI_1.3)</code> [CXXABI]
<code>__cxa_vec_dtor(CXXABI_1.3)</code> [CXXABI]
<code>__cxa_vec_new(CXXABI_1.3)</code> [CXXABI]
<code>__cxa_vec_new2(CXXABI_1.3)</code> [CXXABI]
<code>__cxa_vec_new3(CXXABI_1.3)</code> [CXXABI]
<code>__dynamic_cast(CXXABI_1.3)</code> [CXXABI]
<code>__gxx_personality_v0(CXXABI_1.3)</code> [CXXABI]

An LSB conforming implementation shall provide the generic data interfaces for C++ Runtime Support specified in Table 9-3, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-3 libstdc++ - C++ Runtime Support Data Interfaces**

<code>cin(GLIBCXX_3.4)</code> [ISOCXX]
<code>cerr(GLIBCXX_3.4)</code> [ISOCXX]
<code>clog(GLIBCXX_3.4)</code> [ISOCXX]
<code>cout(GLIBCXX_3.4)</code> [ISOCXX]
<code>wcin(GLIBCXX_3.4)</code> [ISOCXX]
<code>wcerr(GLIBCXX_3.4)</code> [ISOCXX]
<code>wclog(GLIBCXX_3.4)</code> [ISOCXX]
<code>wcout(GLIBCXX_3.4)</code> [ISOCXX]
<code>nothrow(GLIBCXX_3.4)</code> [ISOCXX]

## 9.1.2 C++ type descriptors for built-in types

### 9.1.2.1 Interfaces for C++ type descriptors for built-in types

No external methods are defined for libstdc++ - C++ type descriptors for built-in types in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for C++ type descriptors for built-in types specified in Table 9-4, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-4 libstdc++ - C++ type descriptors for built-in types Data Interfaces**

<code>typeid for signed char const*(CXXABI_1.3)</code> [CXXABI]
<code>typeid for bool const*(CXXABI_1.3)</code> [CXXABI]
<code>typeid for char const*(CXXABI_1.3)</code> [CXXABI]
<code>typeid for double const*(CXXABI_1.3)</code> [CXXABI]
<code>typeid for long double const*(CXXABI_1.3)</code> [CXXABI]

typeinfo for float const*(CXXABI_1.3) [CXXABI]
typeinfo for unsigned char const*(CXXABI_1.3) [CXXABI]
typeinfo for int const*(CXXABI_1.3) [CXXABI]
typeinfo for unsigned int const*(CXXABI_1.3) [CXXABI]
typeinfo for long const*(CXXABI_1.3) [CXXABI]
typeinfo for unsigned long const*(CXXABI_1.3) [CXXABI]
typeinfo for short const*(CXXABI_1.3) [CXXABI]
typeinfo for unsigned short const*(CXXABI_1.3) [CXXABI]
typeinfo for void const*(CXXABI_1.3) [CXXABI]
typeinfo for wchar_t const*(CXXABI_1.3) [CXXABI]
typeinfo for long long const*(CXXABI_1.3) [CXXABI]
typeinfo for unsigned long long const*(CXXABI_1.3) [CXXABI]
typeinfo for signed char*(CXXABI_1.3) [CXXABI]
typeinfo for bool*(CXXABI_1.3) [CXXABI]
typeinfo for char*(CXXABI_1.3) [CXXABI]
typeinfo for double*(CXXABI_1.3) [CXXABI]
typeinfo for long double*(CXXABI_1.3) [CXXABI]
typeinfo for float*(CXXABI_1.3) [CXXABI]
typeinfo for unsigned char*(CXXABI_1.3) [CXXABI]
typeinfo for int*(CXXABI_1.3) [CXXABI]
typeinfo for unsigned int*(CXXABI_1.3) [CXXABI]
typeinfo for long*(CXXABI_1.3) [CXXABI]
typeinfo for unsigned long*(CXXABI_1.3) [CXXABI]
typeinfo for short*(CXXABI_1.3) [CXXABI]
typeinfo for unsigned short*(CXXABI_1.3) [CXXABI]
typeinfo for void*(CXXABI_1.3) [CXXABI]
typeinfo for wchar_t*(CXXABI_1.3) [CXXABI]
typeinfo for long long*(CXXABI_1.3) [CXXABI]
typeinfo for unsigned long long*(CXXABI_1.3) [CXXABI]
typeinfo for signed char(CXXABI_1.3) [CXXABI]
typeinfo for bool(CXXABI_1.3) [CXXABI]
typeinfo for char(CXXABI_1.3) [CXXABI]
typeinfo for double(CXXABI_1.3) [CXXABI]
typeinfo for long double(CXXABI_1.3) [CXXABI]

typeinfo for float(CXXABI_1.3) [CXXABI]
typeinfo for unsigned char(CXXABI_1.3) [CXXABI]
typeinfo for int(CXXABI_1.3) [CXXABI]
typeinfo for unsigned int(CXXABI_1.3) [CXXABI]
typeinfo for long(CXXABI_1.3) [CXXABI]
typeinfo for unsigned long(CXXABI_1.3) [CXXABI]
typeinfo for short(CXXABI_1.3) [CXXABI]
typeinfo for unsigned short(CXXABI_1.3) [CXXABI]
typeinfo for void(CXXABI_1.3) [CXXABI]
typeinfo for wchar_t(CXXABI_1.3) [CXXABI]
typeinfo for long long(CXXABI_1.3) [CXXABI]
typeinfo for unsigned long long(CXXABI_1.3) [CXXABI]
typeinfo name for signed char const*(CXXABI_1.3) [CXXABI]
typeinfo name for bool const*(CXXABI_1.3) [CXXABI]
typeinfo name for char const*(CXXABI_1.3) [CXXABI]
typeinfo name for double const*(CXXABI_1.3) [CXXABI]
typeinfo name for long double const*(CXXABI_1.3) [CXXABI]
typeinfo name for float const*(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned char const*(CXXABI_1.3) [CXXABI]
typeinfo name for int const*(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned int const*(CXXABI_1.3) [CXXABI]
typeinfo name for long const*(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned long const*(CXXABI_1.3) [CXXABI]
typeinfo name for short const*(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned short const*(CXXABI_1.3) [CXXABI]
typeinfo name for void const*(CXXABI_1.3) [CXXABI]
typeinfo name for wchar_t const*(CXXABI_1.3) [CXXABI]
typeinfo name for long long const*(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned long long const*(CXXABI_1.3) [CXXABI]
typeinfo name for signed char*(CXXABI_1.3) [CXXABI]
typeinfo name for bool*(CXXABI_1.3) [CXXABI]
typeinfo name for char*(CXXABI_1.3) [CXXABI]
typeinfo name for double*(CXXABI_1.3) [CXXABI]
typeinfo name for long double*(CXXABI_1.3) [CXXABI]

typeinfo name for float*(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned char*(CXXABI_1.3) [CXXABI]
typeinfo name for int*(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned int*(CXXABI_1.3) [CXXABI]
typeinfo name for long*(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned long*(CXXABI_1.3) [CXXABI]
typeinfo name for short*(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned short*(CXXABI_1.3) [CXXABI]
typeinfo name for void*(CXXABI_1.3) [CXXABI]
typeinfo name for wchar_t*(CXXABI_1.3) [CXXABI]
typeinfo name for long long*(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned long long*(CXXABI_1.3) [CXXABI]
typeinfo name for signed char(CXXABI_1.3) [CXXABI]
typeinfo name for bool(CXXABI_1.3) [CXXABI]
typeinfo name for char(CXXABI_1.3) [CXXABI]
typeinfo name for double(CXXABI_1.3) [CXXABI]
typeinfo name for long double(CXXABI_1.3) [CXXABI]
typeinfo name for float(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned char(CXXABI_1.3) [CXXABI]
typeinfo name for int(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned int(CXXABI_1.3) [CXXABI]
typeinfo name for long(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned long(CXXABI_1.3) [CXXABI]
typeinfo name for short(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned short(CXXABI_1.3) [CXXABI]
typeinfo name for void(CXXABI_1.3) [CXXABI]
typeinfo name for wchar_t(CXXABI_1.3) [CXXABI]
typeinfo name for long long(CXXABI_1.3) [CXXABI]
typeinfo name for unsigned long long(CXXABI_1.3) [CXXABI]

### 9.1.3 C++ `_Rb_tree`

#### 9.1.3.1 Interfaces for C++ `_Rb_tree`

An LSB conforming implementation shall provide the generic methods for C++ `_Rb_tree` specified in Table 9-5, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-5 libstdc++ - C++ `_Rb_tree` Function Interfaces**

<code>_Rb_tree_decrement(_Rb_tree_node_base const*)(GLIBCXX_3.4) [LSB]</code>
<code>_Rb_tree_decrement(_Rb_tree_node_base*)(GLIBCXX_3.4) [LSB]</code>
<code>_Rb_tree_increment(_Rb_tree_node_base const*)(GLIBCXX_3.4) [LSB]</code>
<code>_Rb_tree_increment(_Rb_tree_node_base*)(GLIBCXX_3.4) [LSB]</code>
<code>_Rb_tree_black_count(_Rb_tree_node_base const*, _Rb_tree_node_base const*)(GLIBCXX_3.4) [LSB]</code>
<code>_Rb_tree_rotate_left(_Rb_tree_node_base*, _Rb_tree_node_base*&amp;)(GLIBCXX_3.4) [LSB]</code>
<code>_Rb_tree_rotate_right(_Rb_tree_node_base*, _Rb_tree_node_base*&amp;)(GLIBCXX_3.4) [LSB]</code>
<code>_Rb_tree_rebalance_for_erase(_Rb_tree_node_base*, _Rb_tree_node_base&amp;)(GLIBCXX_3.4) [LSB]</code>
<code>_Rb_tree_insert_and_rebalance(bool, _Rb_tree_node_base*, _Rb_tree_node_base*, _Rb_tree_node_base&amp;)(GLIBCXX_3.4) [LSB]</code>

## 9.1.4 Class `type_info`

### 9.1.4.1 Class data for `type_info`

The virtual table for the `std::type_info` class is described by Table 9-6

**Table 9-6 Primary vtable for `type_info`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeinfo</code> for <code>type_info</code>
<code>vfunc[0]:</code>	<code>type_info::~~type_info()</code>
<code>vfunc[1]:</code>	<code>type_info::~~type_info()</code>
<code>vfunc[2]:</code>	<code>type_info::__is_pointer_p() const</code>
<code>vfunc[3]:</code>	<code>type_info::__is_function_p() const</code>
<code>vfunc[4]:</code>	<code>type_info::__do_catch(type_info const*, void**, unsigned int) const</code>
<code>vfunc[5]:</code>	<code>type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</code>

The Run Time Type Information for the `std::type_info` class is described by Table 9-7

**Table 9-7 `typeinfo` for `type_info`**

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

### 9.1.4.2 Interfaces for Class `type_info`

An LSB conforming implementation shall provide the generic methods for Class `std::type_info` specified in Table 9-8, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-8 libstdcxx - Class `type_info` Function Interfaces**

<code>type_info::__do_catch(type_info const*, void**, unsigned int)</code> const(GLIBCXX_3.4) [ISOCXX]
<code>type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**)</code> const(GLIBCXX_3.4) [ISOCXX]
<code>type_info::__is_pointer_p()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>type_info::__is_function_p()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>type_info::~type_info()</code> (GLIBCXX_3.4) [ISOCXX]
<code>type_info::~type_info()</code> (GLIBCXX_3.4) [ISOCXX]
<code>type_info::~type_info()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::type_info` specified in Table 9-9, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-9 libstdcxx - Class `type_info` Data Interfaces**

<code>typeid</code> for <code>type_info</code> (GLIBCXX_3.4) [CXXABI]
<code>typeid</code> name for <code>type_info</code> (GLIBCXX_3.4) [CXXABI]
<code>vtable</code> for <code>type_info</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.5 Class `__cxxabiv1::__enum_type_info`

#### 9.1.5.1 Class data for `__cxxabiv1::__enum_type_info`

The virtual table for the `__cxxabiv1::__enum_type_info` class is described by Table 9-10

**Table 9-10 Primary vtable for `__cxxabiv1::__enum_type_info`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid</code> for <code>__cxxabiv1::__enum_type_info</code>
<code>vfunc[0]:</code>	<code>__cxxabiv1::__enum_type_info::~__enum_type_info()</code>
<code>vfunc[1]:</code>	<code>__cxxabiv1::__enum_type_info::~__enum_type_info()</code>
<code>vfunc[2]:</code>	<code>type_info::__is_pointer_p()</code> const
<code>vfunc[3]:</code>	<code>type_info::__is_function_p()</code> const
<code>vfunc[4]:</code>	<code>type_info::__do_catch(type_info</code>

	const*, void**, unsigned int) const
vfunc[5]:	type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const

The Run Time Type Information for the `__cxxabiv1::__enum_type_info` class is described by Table 9-11

**Table 9-11 typeinfo for `__cxxabiv1::__enum_type_info`**

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

### 9.1.5.2 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 9-12, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-12 libstdcxx - Class `__cxxabiv1::__enum_type_info` Function Interfaces**

<code>__cxxabiv1::__enum_type_info::~__enum_type_info()(CXXABI_1.3)</code> [CXXABI]
<code>__cxxabiv1::__enum_type_info::~__enum_type_info()(CXXABI_1.3)</code> [CXXABI]
<code>__cxxabiv1::__enum_type_info::~__enum_type_info()(CXXABI_1.3)</code> [CXXABI]

An LSB conforming implementation shall provide the generic data interfaces for Class `__cxxabiv1::__enum_type_info` specified in Table 9-13, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-13 libstdcxx - Class `__cxxabiv1::__enum_type_info` Data Interfaces**

typeinfo for <code>__cxxabiv1::__enum_type_info(CXXABI_1.3)</code> [CXXABI]
typeinfo name for <code>__cxxabiv1::__enum_type_info(CXXABI_1.3)</code> [CXXABI]
vtable for <code>__cxxabiv1::__enum_type_info(CXXABI_1.3)</code> [CXXABI]

### 9.1.6 Class `__cxxabiv1::__array_type_info`

#### 9.1.6.1 Class data for `__cxxabiv1::__array_type_info`

The virtual table for the `__cxxabiv1::__array_type_info` class is described by Table 9-14

**Table 9-14 Primary vtable for `__cxxabiv1::__array_type_info`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for

	<code>__cxxabiv1::__array_type_info</code>
<code>vfunc[0]:</code>	<code>__cxxabiv1::__array_type_info::~~__array_type_info()</code>
<code>vfunc[1]:</code>	<code>__cxxabiv1::__array_type_info::~~__array_type_info()</code>
<code>vfunc[2]:</code>	<code>type_info::__is_pointer_p() const</code>
<code>vfunc[3]:</code>	<code>type_info::__is_function_p() const</code>
<code>vfunc[4]:</code>	<code>type_info::__do_catch(type_info const*, void**, unsigned int) const</code>
<code>vfunc[5]:</code>	<code>type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</code>

The Run Time Type Information for the `__cxxabiv1::__array_type_info` class is described by Table 9-15

**Table 9-15 typeinfo for `__cxxabiv1::__array_type_info`**

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

### 9.1.6.2 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 9-16, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-16 libstdcxx - Class `__cxxabiv1::__array_type_info` Function Interfaces**

<code>__cxxabiv1::__array_type_info::~~__array_type_info()(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__array_type_info::~~__array_type_info()(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__array_type_info::~~__array_type_info()(CXXABI_1.3) [CXXABI]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `__cxxabiv1::__array_type_info` specified in Table 9-17, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-17 libstdcxx - Class `__cxxabiv1::__array_type_info` Data Interfaces**

typeinfo for <code>__cxxabiv1::__array_type_info(CXXABI_1.3) [CXXABI]</code>
typeinfo name for <code>__cxxabiv1::__array_type_info(CXXABI_1.3) [CXXABI]</code>
vtable for <code>__cxxabiv1::__array_type_info(CXXABI_1.3) [CXXABI]</code>

### 9.1.7 Class `__cxxabiv1::__class_type_info`

#### 9.1.7.1 Class data for `__cxxabiv1::__class_type_info`

The virtual table for the `__cxxabiv1::__class_type_info` class is described by Table 9-18



**Table 9-18 Primary vtable for `__cxxabiv1::__class_type_info`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>__cxxabiv1::__class_type_info</code>
<code>vfunc[0]:</code>	<code>__cxxabiv1::__class_type_info::~~class_type_info()</code>
<code>vfunc[1]:</code>	<code>__cxxabiv1::__class_type_info::~~class_type_info()</code>
<code>vfunc[2]:</code>	<code>type_info::__is_pointer_p() const</code>
<code>vfunc[3]:</code>	<code>type_info::__is_function_p() const</code>
<code>vfunc[4]:</code>	<code>__cxxabiv1::__class_type_info::__do_catch(type_info const*, void**, unsigned int) const</code>
<code>vfunc[5]:</code>	<code>__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</code>
<code>vfunc[6]:</code>	<code>__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&amp;) const</code>

The Run Time Type Information for the `__cxxabiv1::__class_type_info` class is described by Table 9-19

**Table 9-19 typeid for `__cxxabiv1::__class_type_info`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>__cxxabiv1::__class_type_info</code>

### 9.1.7.2 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 9-20, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-20 libstdc++ - Class `__cxxabiv1::__class_type_info` Function Interfaces**

<code>__cxxabiv1::__class_type_info::~~class_type_info()(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__class_type_info::~~class_type_info()(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__class_type_info::~~class_type_info()(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__class_type_info::__do_catch(type_info const*, void**, unsigned int) const(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info</code>

<code>const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&amp;)</code> <code>const(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info</code> <code>const*, void**) const(CXXABI_1.3) [CXXABI]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `__cxxabiv1::__class_type_info` specified in Table 9-21, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-21 libstdc++ - Class `__cxxabiv1::__class_type_info` Data Interfaces**

<code>typeinfo for __cxxabiv1::__class_type_info(CXXABI_1.3) [CXXABI]</code>
<code>typeinfo name for __cxxabiv1::__class_type_info(CXXABI_1.3) [CXXABI]</code>
<code>vtable for __cxxabiv1::__class_type_info(CXXABI_1.3) [CXXABI]</code>

## 9.1.8 Class `__cxxabiv1::__pbase_type_info`

### 9.1.8.1 Class data for `__cxxabiv1::__pbase_type_info`

The virtual table for the `__cxxabiv1::__pbase_type_info` class is described by Table 9-22

**Table 9-22 Primary vtable for `__cxxabiv1::__pbase_type_info`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeinfo for</code> <code>__cxxabiv1::__pbase_type_info</code>
<code>vfunc[0]:</code>	<code>__cxxabiv1::__pbase_type_info::~~_p</code> <code>base_type_info()</code>
<code>vfunc[1]:</code>	<code>__cxxabiv1::__pbase_type_info::~~_p</code> <code>base_type_info()</code>
<code>vfunc[2]:</code>	<code>type_info::__is_pointer_p() const</code>
<code>vfunc[3]:</code>	<code>type_info::__is_function_p() const</code>
<code>vfunc[4]:</code>	<code>__cxxabiv1::__pbase_type_info::__do</code> <code>_catch(type_info const*, void**, unsigned int) const</code>
<code>vfunc[5]:</code>	<code>type_info::__do_upcast(__cxxabiv1::__</code> <code>_class_type_info const*, void**) const</code>
<code>vfunc[6]:</code>	<code>__cxxabiv1::__pbase_type_info::__poi</code> <code>nter_catch(__cxxabiv1::__pbase_type</code> <code>_info const*, void**, unsigned int)</code> <code>const</code>

The Run Time Type Information for the `__cxxabiv1::__pbase_type_info` class is described by Table 9-23

Table 9-23 typeinfo for `__cxxabiv1::__pbase_type_info`

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

### 9.1.8.2 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 9-24, with the full mandatory functionality as described in the referenced underlying specification.

Table 9-24 `libstdc++` - Class `__cxxabiv1::__pbase_type_info` Function Interfaces

<code>__cxxabiv1::__pbase_type_info::~__pbase_type_info()(CXXABI_1.3)</code> [CXXABI]
<code>__cxxabiv1::__pbase_type_info::~__pbase_type_info()(CXXABI_1.3)</code> [CXXABI]
<code>__cxxabiv1::__pbase_type_info::~__pbase_type_info()(CXXABI_1.3)</code> [CXXABI]
<code>__cxxabiv1::__pbase_type_info::~do_catch(type_info const*, void**, unsigned int) const(CXXABI_1.3)</code> [CXXABI]
<code>__cxxabiv1::__pbase_type_info::~pointer_catch(__cxxabiv1::__pbase_type_info const*, void**, unsigned int) const(CXXABI_1.3)</code> [CXXABI]

An LSB conforming implementation shall provide the generic data interfaces for Class `__cxxabiv1::__pbase_type_info` specified in Table 9-25, with the full mandatory functionality as described in the referenced underlying specification.

Table 9-25 `libstdc++` - Class `__cxxabiv1::__pbase_type_info` Data Interfaces

typeinfo for <code>__cxxabiv1::__pbase_type_info(CXXABI_1.3)</code> [CXXABI]
typeinfo name for <code>__cxxabiv1::__pbase_type_info(CXXABI_1.3)</code> [CXXABI]
vtable for <code>__cxxabiv1::__pbase_type_info(CXXABI_1.3)</code> [CXXABI]

### 9.1.9 Class `__cxxabiv1::__pointer_type_info`

#### 9.1.9.1 Class data for `__cxxabiv1::__pointer_type_info`

The virtual table for the `__cxxabiv1::__pointer_type_info` class is described by Table 9-26

Table 9-26 Primary vtable for `__cxxabiv1::__pointer_type_info`

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>__cxxabiv1::__pointer_type_info</code>
<code>vfunc[0]:</code>	<code>__cxxabiv1::__pointer_type_info::~__</code>

	pointer_type_info()
vfunc[1]:	__cxxabiv1::__pointer_type_info::~~__pointer_type_info()
vfunc[2]:	__cxxabiv1::__pointer_type_info::__is__pointer_p() const
vfunc[3]:	type_info::__is_function_p() const
vfunc[4]:	__cxxabiv1::__pbase_type_info::__do__catch(type_info const*, void**, unsigned int) const
vfunc[5]:	type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const
vfunc[6]:	__cxxabiv1::__pointer_type_info::__p__pointer_catch(__cxxabiv1::__pbase_type_info const*, void**, unsigned int) const

The Run Time Type Information for the `__cxxabiv1::__pointer_type_info` class is described by Table 9-27

**Table 9-27 typeinfo for `__cxxabiv1::__pointer_type_info`**

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

### 9.1.9.2 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 9-28, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-28 libstdc++ - Class `__cxxabiv1::__pointer_type_info` Function Interfaces**

<code>__cxxabiv1::__pointer_type_info::~~__pointer_type_info()</code> (CXXABI_1.3) [CXXABI]
<code>__cxxabiv1::__pointer_type_info::~~__pointer_type_info()</code> (CXXABI_1.3) [CXXABI]
<code>__cxxabiv1::__pointer_type_info::~~__pointer_type_info()</code> (CXXABI_1.3) [CXXABI]
<code>__cxxabiv1::__pointer_type_info::__is_pointer_p() const</code> (CXXABI_1.3) [CXXABI]
<code>__cxxabiv1::__pointer_type_info::__pointer_catch(__cxxabiv1::__pbase_type_info const*, void**, unsigned int) const</code> (CXXABI_1.3) [CXXABI]

An LSB conforming implementation shall provide the generic data interfaces for Class `__cxxabiv1::__pointer_type_info` specified in Table 9-29, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-29 libstdcxx - Class `__cxxabiv1::__pointer_type_info` Data Interfaces**

typeinfo for <code>__cxxabiv1::__pointer_type_info(CXXABI_1.3)</code> [CXXABI]
typeinfo name for <code>__cxxabiv1::__pointer_type_info(CXXABI_1.3)</code> [CXXABI]
vtable for <code>__cxxabiv1::__pointer_type_info(CXXABI_1.3)</code> [CXXABI]

### 9.1.10 Class `__cxxabiv1::__function_type_info`

#### 9.1.10.1 Class data for `__cxxabiv1::__function_type_info`

The virtual table for the `__cxxabiv1::__function_type_info` class is described by Table 9-30

**Table 9-30 Primary vtable for `__cxxabiv1::__function_type_info`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>__cxxabiv1::__function_type_info</code>
vfunc[0]:	<code>__cxxabiv1::__function_type_info::~~ __function_type_info()</code>
vfunc[1]:	<code>__cxxabiv1::__function_type_info::~~ __function_type_info()</code>
vfunc[2]:	<code>type_info::__is_pointer_p() const</code>
vfunc[3]:	<code>__cxxabiv1::__function_type_info::__i s_function_p() const</code>
vfunc[4]:	<code>type_info::__do_catch(type_info const*, void**, unsigned int) const</code>
vfunc[5]:	<code>type_info::__do_upcast(__cxxabiv1:: __class_type_info const*, void**) const</code>

The Run Time Type Information for the `__cxxabiv1::__function_type_info` class is described by Table 9-31

**Table 9-31 typeinfo for `__cxxabiv1::__function_type_info`**

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

#### 9.1.10.2 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 9-32, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-32 libstdcxx - Class `__cxxabiv1::__function_type_info` Function Interfaces**

<code>__cxxabiv1::__function_type_info::~~ __function_type_info()</code> (CXXABI_1.3)
---

[CXXABI]
<code>__cxxabiv1::__function_type_info::~__function_type_info()</code> (CXXABI_1.3) [CXXABI]
<code>__cxxabiv1::__function_type_info::~__function_type_info()</code> (CXXABI_1.3) [CXXABI]
<code>__cxxabiv1::__function_type_info::__is_function_p()</code> const(CXXABI_1.3) [CXXABI]

An LSB conforming implementation shall provide the generic data interfaces for Class `__cxxabiv1::__function_type_info` specified in Table 9-33, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-33 libstdc++ - Class `__cxxabiv1::__function_type_info` Data Interfaces**

<code>typeid</code> for <code>__cxxabiv1::__function_type_info</code> (CXXABI_1.3) [CXXABI]
<code>typeid</code> name for <code>__cxxabiv1::__function_type_info</code> (CXXABI_1.3) [CXXABI]
<code>vtable</code> for <code>__cxxabiv1::__function_type_info</code> (CXXABI_1.3) [CXXABI]

### 9.1.11 Class `__cxxabiv1::__si_class_type_info`

#### 9.1.11.1 Class data for `__cxxabiv1::__si_class_type_info`

The virtual table for the `__cxxabiv1::__si_class_type_info` class is described by Table 9-34

**Table 9-34 Primary vtable for `__cxxabiv1::__si_class_type_info`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid</code> for <code>__cxxabiv1::__si_class_type_info</code>
<code>vfunc[0]:</code>	<code>__cxxabiv1::__si_class_type_info::~__si_class_type_info()</code>
<code>vfunc[1]:</code>	<code>__cxxabiv1::__si_class_type_info::~__si_class_type_info()</code>
<code>vfunc[2]:</code>	<code>type_info::__is_pointer_p()</code> const
<code>vfunc[3]:</code>	<code>type_info::__is_function_p()</code> const
<code>vfunc[4]:</code>	<code>__cxxabiv1::__class_type_info::__do_catch</code> ( <code>type_info</code> const*, void**, unsigned int) const
<code>vfunc[5]:</code>	<code>__cxxabiv1::__class_type_info::__do_upcast</code> ( <code>__cxxabiv1::__class_type_info</code> const*, void**) const
<code>vfunc[6]:</code>	<code>__cxxabiv1::__si_class_type_info::__do_upcast</code> ( <code>__cxxabiv1::__class_type_info</code> const*, void const*, <code>__cxxabiv1::__class_type_info::__upc</code>

	ast_result&) const
--	--------------------

The Run Time Type Information for the `__cxxabiv1::__si_class_type_info` class is described by Table 9-35

**Table 9-35 typeinfo for `__cxxabiv1::__si_class_type_info`**

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

### 9.1.11.2 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 9-36, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-36 libstdc++ - Class `__cxxabiv1::__si_class_type_info` Function Interfaces**

<code>__cxxabiv1::__si_class_type_info::~~__si_class_type_info()(CXXABI_1.3)</code> [CXXABI]
<code>__cxxabiv1::__si_class_type_info::~~__si_class_type_info()(CXXABI_1.3)</code> [CXXABI]
<code>__cxxabiv1::__si_class_type_info::~~__si_class_type_info()(CXXABI_1.3)</code> [CXXABI]
<code>__cxxabiv1::__si_class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&amp;) const(CXXABI_1.3)</code> [CXXABI]

An LSB conforming implementation shall provide the generic data interfaces for Class `__cxxabiv1::__si_class_type_info` specified in Table 9-37, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-37 libstdc++ - Class `__cxxabiv1::__si_class_type_info` Data Interfaces**

typeinfo for <code>__cxxabiv1::__si_class_type_info(CXXABI_1.3)</code> [CXXABI]
typeinfo name for <code>__cxxabiv1::__si_class_type_info(CXXABI_1.3)</code> [CXXABI]
vtable for <code>__cxxabiv1::__si_class_type_info(CXXABI_1.3)</code> [CXXABI]

### 9.1.12 Class `__cxxabiv1::__vmi_class_type_info`

#### 9.1.12.1 Class data for `__cxxabiv1::__vmi_class_type_info`

The virtual table for the `__cxxabiv1::__vmi_class_type_info` class is described by Table 9-38

**Table 9-38 Primary vtable for `__cxxabiv1::__vmi_class_type_info`**

Base Offset	0
Virtual Base Offset	0

RTTI	typeinfo for __cxxabiv1::__vmi_class_type_info
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_ catch(type_info const*, void**, unsigned int) const
vfunc[5]:	__cxxabiv1::__class_type_info::__do_ upcast(__cxxabiv1::__class_type_info const*, void**) const
vfunc[6]:	__cxxabiv1::__vmi_class_type_info::__ do_upcast(__cxxabiv1::__class_type _info const*, void const*, __cxxabiv1::__class_type_info::__upc ast_result&) const

The Run Time Type Information for the \_\_cxxabiv1::\_\_vmi\_class\_type\_info class is described by Table 9-39

**Table 9-39 typeinfo for \_\_cxxabiv1::\_\_vmi\_class\_type\_info**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for __cxxabiv1::__vmi_class_type_info

### 9.1.12.2 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 9-40, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-40 libstdcxx - Class \_\_cxxabiv1::\_\_vmi\_class\_type\_info Function Interfaces**

__cxxabiv1::__vmi_class_type_info::~~ __vmi_class_type_info()(CXXABI_1.3) [CXXABI]
__cxxabiv1::__vmi_class_type_info::~~ __vmi_class_type_info()(CXXABI_1.3) [CXXABI]
__cxxabiv1::__vmi_class_type_info::~~ __vmi_class_type_info()(CXXABI_1.3) [CXXABI]
__cxxabiv1::__vmi_class_type_info::__do_upcast( __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::: upcast_result&) const(CXXABI_1.3) [CXXABI]



An LSB conforming implementation shall provide the generic data interfaces for Class `__cxxabiv1::__vmi_class_type_info` specified in Table 9-41, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-41 libstdcxx - Class `__cxxabiv1::__vmi_class_type_info` Data Interfaces**

typeinfo for <code>__cxxabiv1::__vmi_class_type_info(CXXABI_1.3)</code> [CXXABI]
typeinfo name for <code>__cxxabiv1::__vmi_class_type_info(CXXABI_1.3)</code> [CXXABI]
vtable for <code>__cxxabiv1::__vmi_class_type_info(CXXABI_1.3)</code> [CXXABI]

### 9.1.13 Class `__cxxabiv1::__fundamental_type_info`

#### 9.1.13.1 Class data for `__cxxabiv1::__fundamental_type_info`

The virtual table for the `__cxxabiv1::__fundamental_type_info` class is described by Table 9-42

**Table 9-42 Primary vtable for `__cxxabiv1::__fundamental_type_info`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>__cxxabiv1::__fundamental_type_info</code>
vfunc[0]:	<code>__cxxabiv1::__fundamental_type_info::~~__fundamental_type_info()</code>
vfunc[1]:	<code>__cxxabiv1::__fundamental_type_info::~~__fundamental_type_info()</code>
vfunc[2]:	<code>type_info::__is_pointer_p() const</code>
vfunc[3]:	<code>type_info::__is_function_p() const</code>
vfunc[4]:	<code>type_info::__do_catch(type_info const*, void**, unsigned int) const</code>
vfunc[5]:	<code>type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</code>

The Run Time Type Information for the `__cxxabiv1::__fundamental_type_info` class is described by Table 9-43

**Table 9-43 typeinfo for `__cxxabiv1::__fundamental_type_info`**

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

### 9.1.13.2 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 9-44, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-44 libstdcxx - Class `__cxxabiv1::__fundamental_type_info` Function Interfaces**

<code>__cxxabiv1::__fundamental_type_info::~~__fundamental_type_info()(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__fundamental_type_info::~~__fundamental_type_info()(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__fundamental_type_info::~~__fundamental_type_info()(CXXABI_1.3) [CXXABI]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `__cxxabiv1::__fundamental_type_info` specified in Table 9-45, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-45 libstdcxx - Class `__cxxabiv1::__fundamental_type_info` Data Interfaces**

<code>typeid for __cxxabiv1::__fundamental_type_info(CXXABI_1.3) [CXXABI]</code>
<code>typeid name for __cxxabiv1::__fundamental_type_info(CXXABI_1.3) [CXXABI]</code>
<code>vtable for __cxxabiv1::__fundamental_type_info(CXXABI_1.3) [CXXABI]</code>

### 9.1.14 Class `__cxxabiv1::__pointer_to_member_type_info`

#### 9.1.14.1 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 9-46

**Table 9-46 Primary vtable for `__cxxabiv1::__pointer_to_member_type_info`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid for __cxxabiv1::__pointer_to_member_type_info</code>
<code>vfunc[0]:</code>	<code>__cxxabiv1::__pointer_to_member_type_info::~~__pointer_to_member_type_info()</code>
<code>vfunc[1]:</code>	<code>__cxxabiv1::__pointer_to_member_type_info::~~__pointer_to_member_type_info()</code>
<code>vfunc[2]:</code>	<code>type_info::__is_pointer_p() const</code>
<code>vfunc[3]:</code>	<code>type_info::__is_function_p() const</code>

vfunc[4]:	<code>__cxxabiv1::__pbase_type_info::__do_catch(type_info const*, void**, unsigned int) const</code>
vfunc[5]:	<code>type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const</code>
vfunc[6]:	<code>__cxxabiv1::__pointer_to_member_type_info::__pointer_catch(__cxxabiv1::__pbase_type_info const*, void**, unsigned int) const</code>

The Run Time Type Information for the `__cxxabiv1::__pointer_to_member_type_info` class is described by Table 9-47

**Table 9-47 typeinfo for `__cxxabiv1::__pointer_to_member_type_info`**

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

#### 9.1.14.2 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 9-48, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-48 libstdcxx - Class `__cxxabiv1::__pointer_to_member_type_info` Function Interfaces**

<code>__cxxabiv1::__pointer_to_member_type_info::~__pointer_to_member_type_info()(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__pointer_to_member_type_info::~__pointer_to_member_type_info()(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__pointer_to_member_type_info::~__pointer_to_member_type_info()(CXXABI_1.3) [CXXABI]</code>
<code>__cxxabiv1::__pointer_to_member_type_info::__pointer_catch(__cxxabiv1::__pbase_type_info const*, void**, unsigned int) const(CXXABI_1.3) [CXXABI]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `__cxxabiv1::__pointer_to_member_type_info` specified in Table 9-49, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-49 libstdcxx - Class `__cxxabiv1::__pointer_to_member_type_info` Data Interfaces**

typeinfo for <code>__cxxabiv1::__pointer_to_member_type_info(CXXABI_1.3)</code> [CXXABI]
typeinfo name for <code>__cxxabiv1::__pointer_to_member_type_info(CXXABI_1.3)</code>

[CXXABI]
vtable for <code>__cxxabiv1::__pointer_to_member_type_info(CXXABI_1.3)</code> [CXXABI]

### 9.1.15 Class `__gnu_cxx::stdio_filebuf<char, char_traits<char>>`

#### 9.1.15.1 Class data for `__gnu_cxx::stdio_filebuf<char, char_traits<char>>`

The virtual table for the `__gnu_cxx::stdio_filebuf<char, std::char_traits<char>>` class is described by Table 9-50

**Table 9-50 Primary vtable for `__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt;&gt;</code>
<code>vfunc[0]:</code>	Unspecified
<code>vfunc[1]:</code>	Unspecified
<code>vfunc[2]:</code>	<code>basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;::imbue(locale const&amp;)</code>
<code>vfunc[3]:</code>	See The Architecture Specific Specification
<code>vfunc[4]:</code>	Unspecified
<code>vfunc[5]:</code>	Unspecified
<code>vfunc[6]:</code>	Unspecified
<code>vfunc[7]:</code>	<code>basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;::showmanyc()</code>
<code>vfunc[8]:</code>	Unspecified
<code>vfunc[9]:</code>	Unspecified
<code>vfunc[10]:</code>	Unspecified
<code>vfunc[11]:</code>	Unspecified
<code>vfunc[12]:</code>	Unspecified
<code>vfunc[13]:</code>	Unspecified

#### 9.1.15.2 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for Class `__gnu_cxx::stdio_filebuf<char, std::char_traits<char> >` specified in Table 9-51, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-51 libstdc++ - Class `__gnu_cxx::stdio_filebuf<char, char_traits<char> >` Data Interfaces**

typeid for <code>__gnu_cxx::stdio_filebuf&lt;char, char_traits&lt;char&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>__gnu_cxx::stdio_filebuf&lt;char, char_traits&lt;char&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.16 Class `__gnu_cxx::stdio_filebuf<wchar_t, char_traits<wchar_t> >`

#### 9.1.16.1 Class data for `__gnu_cxx::stdio_filebuf<wchar_t, char_traits<wchar_t> >`

The virtual table for the `__gnu_cxx::stdio_filebuf<wchar_t, std::char_traits<wchar_t> >` class is described by Table 9-52

**Table 9-52 Primary vtable for `__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>__gnu_cxx::stdio_sync_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;</code>
vfunc[0]:	Unspecified
vfunc[1]:	Unspecified
vfunc[2]:	<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::imbue(locale const&amp;)</code>
vfunc[3]:	See The Architecture Specific Specification
vfunc[4]:	Unspecified
vfunc[5]:	Unspecified
vfunc[6]:	Unspecified
vfunc[7]:	<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::showmanyc()</code>
vfunc[8]:	Unspecified
vfunc[9]:	Unspecified
vfunc[10]:	Unspecified
vfunc[11]:	Unspecified

vfunc[12]:	Unspecified
vfunc[13]:	Unspecified

### 9.1.16.2 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 ISO/IEC 23360.

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 9-53, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-53** `libstdcxx` - Class `__gnu_cxx::stdio_filebuf<wchar_t, char_traits<wchar_t>>` Data Interfaces

<code>typeid for __gnu_cxx::stdio_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for __gnu_cxx::stdio_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.17 Class `__gnu_cxx::__pool_alloc_base`

#### 9.1.17.1 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 9-54, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-54** `libstdcxx` - Class `__gnu_cxx::__pool_alloc_base` Function Interfaces

<code>__gnu_cxx::__pool_alloc_base::_M_get_mutex()(GLIBCXX_3.4.2) [LSB]</code>
--

### 9.1.18 Class `__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>>`

#### 9.1.18.1 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 9-55, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-55** `libstdcxx` - Class `__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>>` Function Interfaces

<code>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt;&gt;::file()(GLIBCXX_3.4.2) [LSB]</code>
--

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 9-56, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-56 libstdcxx - Class `__gnu_cxx::stdio_sync_filebuf<char, char_traits<char>>` Data Interfaces**

typeinfo for <code>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt;&gt;</code> >(GLIBCXX_3.4) [LSB]
typeinfo name for <code>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt;&gt;</code> >(GLIBCXX_3.4) [LSB]
vtable for <code>__gnu_cxx::stdio_sync_filebuf&lt;char, char_traits&lt;char&gt;&gt;</code> >(GLIBCXX_3.4) [LSB]

### 9.1.19 Class `__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>`

#### 9.1.19.1 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 9-57, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-57 libstdcxx - Class `__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>` Function Interfaces**

<code>__gnu_cxx::stdio_sync_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</code> <code>::file()</code> (GLIBCXX_3.4.2) [LSB]
--

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 9-58, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-58 libstdcxx - Class `__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t>>` Data Interfaces**

typeinfo for <code>__gnu_cxx::stdio_sync_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</code> >(GLIBCXX_3.4) [LSB]
typeinfo name for <code>__gnu_cxx::stdio_sync_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</code> >(GLIBCXX_3.4) [LSB]
vtable for <code>__gnu_cxx::stdio_sync_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</code> >(GLIBCXX_3.4) [LSB]

### 9.1.20 Class exception

#### 9.1.20.1 Class data for exception

The virtual table for the `std::exception` class is described by Table 9-59

**Table 9-59 Primary vtable for exception**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for exception

vfunc[0]:	exception::~~exception()
vfunc[1]:	exception::~~exception()
vfunc[2]:	exception::what() const

The Run Time Type Information for the `std::exception` class is described by Table 9-60

**Table 9-60 typeid for exception**

Base Vtable	vtable for <code>__cxxabiv1::__class_type_info</code>
Name	typeid name for exception

### 9.1.20.2 Interfaces for Class `exception`

An LSB conforming implementation shall provide the generic methods for Class `std::exception` specified in Table 9-61, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-61 libstdc++ - Class `exception` Function Interfaces**

<code>exception::what() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>exception::~~exception()</code> (GLIBCXX_3.4) [ISOCXX]
<code>exception::~~exception()</code> (GLIBCXX_3.4) [ISOCXX]
<code>exception::~~exception()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::exception` specified in Table 9-62, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-62 libstdc++ - Class `exception` Data Interfaces**

typeid for exception(GLIBCXX_3.4) [CXXABI]
typeid name for exception(GLIBCXX_3.4) [CXXABI]
vtable for exception(GLIBCXX_3.4) [CXXABI]

### 9.1.21 Class `bad_typeid`

#### 9.1.21.1 Class data for `bad_typeid`

The virtual table for the `std::bad_typeid` class is described by Table 9-63

**Table 9-63 Primary vtable for `bad_typeid`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>bad_typeid</code>
vfunc[0]:	<code>bad_typeid::~~bad_typeid()</code>
vfunc[1]:	<code>bad_typeid::~~bad_typeid()</code>



vfunc[2]:	exception::what() const
-----------	-------------------------

The Run Time Type Information for the `std::bad_typeid` class is described by Table 9-64

**Table 9-64 typeinfo for bad\_typeid**

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

### 9.1.21.2 Interfaces for Class `bad_typeid`

An LSB conforming implementation shall provide the generic methods for Class `std::bad_typeid` specified in Table 9-65, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-65 libstdcxx - Class `bad_typeid` Function Interfaces**

<code>bad_typeid::~bad_typeid()</code> (GLIBCXX_3.4) [ISOCXX]
<code>bad_typeid::~bad_typeid()</code> (GLIBCXX_3.4) [ISOCXX]
<code>bad_typeid::~bad_typeid()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::bad_typeid` specified in Table 9-66, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-66 libstdcxx - Class `bad_typeid` Data Interfaces**

typeinfo for <code>bad_typeid</code> (GLIBCXX_3.4) [CXXABI]
typeinfo name for <code>bad_typeid</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>bad_typeid</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.22 Class `logic_error`

#### 9.1.22.1 Class data for `logic_error`

The virtual table for the `std::logic_error` class is described by Table 9-67

**Table 9-67 Primary vtable for `logic_error`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>logic_error</code>
vfunc[0]:	<code>logic_error::~logic_error()</code>
vfunc[1]:	<code>logic_error::~logic_error()</code>
vfunc[2]:	<code>logic_error::what() const</code>

The Run Time Type Information for the `std::logic_error` class is described by Table 9-68

**Table 9-68 typeid for logic\_error**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for logic_error

**9.1.22.2 Interfaces for Class logic\_error**

An LSB conforming implementation shall provide the generic methods for Class `std::logic_error` specified in Table 9-69, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-69 libstdc++ - Class logic\_error Function Interfaces**

<code>logic_error::what() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>logic_error::logic_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>logic_error::logic_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>logic_error::~~logic_error()</code> (GLIBCXX_3.4) [ISOCXX]
<code>logic_error::~~logic_error()</code> (GLIBCXX_3.4) [ISOCXX]
<code>logic_error::~~logic_error()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::logic_error` specified in Table 9-70, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-70 libstdc++ - Class logic\_error Data Interfaces**

typeid for <code>logic_error</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>logic_error</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>logic_error</code> (GLIBCXX_3.4) [CXXABI]

**9.1.23 Class range\_error****9.1.23.1 Class data for range\_error**

The virtual table for the `std::range_error` class is described by Table 9-71

**Table 9-71 Primary vtable for range\_error**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>range_error</code>
<code>vfunc[0]:</code>	<code>range_error::~~range_error()</code>
<code>vfunc[1]:</code>	<code>range_error::~~range_error()</code>
<code>vfunc[2]:</code>	<code>runtime_error::what() const</code>

The Run Time Type Information for the `std::range_error` class is described by Table 9-72

**Table 9-72 typeid for range\_error**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>range_error</code>

### 9.1.23.2 Interfaces for Class `range_error`

An LSB conforming implementation shall provide the generic methods for Class `std::range_error` specified in Table 9-73, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-73 libstdc++ - Class `range_error` Function Interfaces**

<code>range_error::range_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>range_error::range_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>range_error::~range_error()(GLIBCXX_3.4) [ISOCXX]</code>
<code>range_error::~range_error()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::range_error` specified in Table 9-74, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-74 libstdc++ - Class `range_error` Data Interfaces**

typeid for <code>range_error</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>range_error</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>range_error</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.24 Class `domain_error`

#### 9.1.24.1 Class data for `domain_error`

The virtual table for the `std::domain_error` class is described by Table 9-75

**Table 9-75 Primary vtable for `domain_error`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>domain_error</code>
<code>vfunc[0]:</code>	<code>domain_error::~domain_error()</code>
<code>vfunc[1]:</code>	<code>domain_error::~domain_error()</code>
<code>vfunc[2]:</code>	<code>logic_error::what() const</code>

The Run Time Type Information for the `std::domain_error` class is described by Table 9-76

**Table 9-76 typeinfo for domain\_error**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for domain_error

**9.1.24.2 Interfaces for Class domain\_error**

An LSB conforming implementation shall provide the generic methods for Class `std::domain_error` specified in Table 9-77, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-77 libstdcxx - Class domain\_error Function Interfaces**

<code>domain_error::domain_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>domain_error::domain_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>domain_error::~domain_error()(GLIBCXX_3.4) [ISOCXX]</code>
<code>domain_error::~domain_error()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::domain_error` specified in Table 9-78, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-78 libstdcxx - Class domain\_error Data Interfaces**

typeinfo for domain_error(GLIBCXX_3.4) [CXXABI]
typeinfo name for domain_error(GLIBCXX_3.4) [CXXABI]
vtable for domain_error(GLIBCXX_3.4) [CXXABI]

**9.1.25 Class length\_error****9.1.25.1 Class data for length\_error**

The virtual table for the `std::length_error` class is described by Table 9-79

**Table 9-79 Primary vtable for length\_error**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for length_error
vfunc[0]:	<code>length_error::~length_error()</code>
vfunc[1]:	<code>length_error::~length_error()</code>
vfunc[2]:	<code>length_error::what() const</code>

The Run Time Type Information for the `std::length_error` class is described by Table 9-80

**Table 9-80 typeinfo for length\_error**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for length_error

**9.1.25.2 Interfaces for Class length\_error**

An LSB conforming implementation shall provide the generic methods for Class `std::length_error` specified in Table 9-81, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-81 libstdc++ - Class length\_error Function Interfaces**

<code>length_error::length_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>length_error::length_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>length_error::~length_error()(GLIBCXX_3.4) [ISOCXX]</code>
<code>length_error::~length_error()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::length_error` specified in Table 9-82, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-82 libstdc++ - Class length\_error Data Interfaces**

typeinfo for length_error(GLIBCXX_3.4) [CXXABI]
typeinfo name for length_error(GLIBCXX_3.4) [CXXABI]
vtable for length_error(GLIBCXX_3.4) [CXXABI]

**9.1.26 Class out\_of\_range****9.1.26.1 Class data for out\_of\_range**

The virtual table for the `std::out_of_range` class is described by Table 9-83

**Table 9-83 Primary vtable for out\_of\_range**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for out_of_range
vfunc[0]:	<code>out_of_range::~out_of_range()</code>
vfunc[1]:	<code>out_of_range::~out_of_range()</code>
vfunc[2]:	<code>logic_error::what() const</code>

The Run Time Type Information for the `std::out_of_range` class is described by Table 9-84

**Table 9-84 typeinfo for out\_of\_range**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for out_of_range

**9.1.26.2 Interfaces for Class out\_of\_range**

An LSB conforming implementation shall provide the generic methods for Class `std::out_of_range` specified in Table 9-85, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-85 libstdc++ - Class out\_of\_range Function Interfaces**

<code>out_of_range::out_of_range(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>out_of_range::out_of_range(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>out_of_range::~out_of_range()(GLIBCXX_3.4) [ISOCXX]</code>
<code>out_of_range::~out_of_range()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::out_of_range` specified in Table 9-86, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-86 libstdc++ - Class out\_of\_range Data Interfaces**

typeinfo for out_of_range(GLIBCXX_3.4) [CXXABI]
typeinfo name for out_of_range(GLIBCXX_3.4) [CXXABI]
vtable for out_of_range(GLIBCXX_3.4) [CXXABI]

**9.1.27 Class bad\_exception****9.1.27.1 Class data for bad\_exception**

The virtual table for the `std::bad_exception` class is described by Table 9-87

**Table 9-87 Primary vtable for bad\_exception**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for bad_exception
vfunc[0]:	<code>bad_exception::~~bad_exception()</code>
vfunc[1]:	<code>bad_exception::~~bad_exception()</code>
vfunc[2]:	<code>exception::what() const</code>

The Run Time Type Information for the `std::bad_exception` class is described by Table 9-88

**Table 9-88 typeinfo for bad\_exception**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for bad_exception

**9.1.27.2 Interfaces for Class bad\_exception**

An LSB conforming implementation shall provide the generic methods for Class `std::bad_exception` specified in Table 9-89, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-89 libstdcxx - Class bad\_exception Function Interfaces**

<code>bad_exception::~bad_exception()(GLIBCXX_3.4) [ISOCXX]</code>
<code>bad_exception::~bad_exception()(GLIBCXX_3.4) [ISOCXX]</code>
<code>bad_exception::~bad_exception()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::bad_exception` specified in Table 9-90, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-90 libstdcxx - Class bad\_exception Data Interfaces**

typeinfo for bad_exception(GLIBCXX_3.4) [CXXABI]
typeinfo name for bad_exception(GLIBCXX_3.4) [CXXABI]
vtable for bad_exception(GLIBCXX_3.4) [CXXABI]

**9.1.28 Class runtime\_error****9.1.28.1 Class data for runtime\_error**

The virtual table for the `std::runtime_error` class is described by Table 9-91

**Table 9-91 Primary vtable for runtime\_error**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for runtime_error
<code>vfunc[0]:</code>	<code>runtime_error::~runtime_error()</code>
<code>vfunc[1]:</code>	<code>runtime_error::~runtime_error()</code>
<code>vfunc[2]:</code>	<code>runtime_error::what() const</code>

The Run Time Type Information for the `std::runtime_error` class is described by Table 9-92

**Table 9-92 typeinfo for runtime\_error**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for runtime_error

### 9.1.28.2 Interfaces for Class `runtime_error`

An LSB conforming implementation shall provide the generic methods for Class `std::runtime_error` specified in Table 9-93, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-93 `libstdcxx` - Class `runtime_error` Function Interfaces**

<code>runtime_error::what() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>runtime_error::runtime_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>runtime_error::runtime_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>runtime_error::~runtime_error()</code> (GLIBCXX_3.4) [ISOCXX]
<code>runtime_error::~runtime_error()</code> (GLIBCXX_3.4) [ISOCXX]
<code>runtime_error::~runtime_error()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::runtime_error` specified in Table 9-94, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-94 `libstdcxx` - Class `runtime_error` Data Interfaces**

<code>typeid for runtime_error</code> (GLIBCXX_3.4) [CXXABI]
<code>typeid name for runtime_error</code> (GLIBCXX_3.4) [CXXABI]
<code>vtable for runtime_error</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.29 Class `overflow_error`

#### 9.1.29.1 Class data for `overflow_error`

The virtual table for the `std::overflow_error` class is described by Table 9-95

**Table 9-95 Primary vtable for `overflow_error`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid for overflow_error</code>
<code>vfunc[0]:</code>	<code>overflow_error::~overflow_error()</code>
<code>vfunc[1]:</code>	<code>overflow_error::~overflow_error()</code>
<code>vfunc[2]:</code>	<code>runtime_error::what() const</code>

The Run Time Type Information for the `std::overflow_error` class is described by Table 9-96

**Table 9-96 `typeid` for `overflow_error`**

Base Vtable	<code>vtable for __cxxabiv1::__si_class_type_info</code>
-------------	--



Name	typeinfo name for overflow_error
------	----------------------------------

### 9.1.29.2 Interfaces for Class overflow\_error

An LSB conforming implementation shall provide the generic methods for Class `std::overflow_error` specified in Table 9-97, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-97 libstdcxx - Class overflow\_error Function Interfaces**

<code>overflow_error::overflow_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>overflow_error::overflow_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>overflow_error::~~overflow_error()(GLIBCXX_3.4) [ISOCXX]</code>
<code>overflow_error::~~overflow_error()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::overflow_error` specified in Table 9-98, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-98 libstdcxx - Class overflow\_error Data Interfaces**

typeinfo for overflow_error(GLIBCXX_3.4) [CXXABI]
typeinfo name for overflow_error(GLIBCXX_3.4) [CXXABI]
vtable for overflow_error(GLIBCXX_3.4) [CXXABI]

### 9.1.30 Class underflow\_error

#### 9.1.30.1 Class data for underflow\_error

The virtual table for the `std::underflow_error` class is described by Table 9-99

**Table 9-99 Primary vtable for underflow\_error**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for underflow_error
vfunc[0]:	<code>underflow_error::~~underflow_error()</code>
vfunc[1]:	<code>underflow_error::~~underflow_error()</code>
vfunc[2]:	<code>runtime_error::what() const</code>

The Run Time Type Information for the `std::underflow_error` class is described by Table 9-100

**Table 9-100 typeinfo for underflow\_error**

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

### 9.1.30.2 Interfaces for Class `underflow_error`

An LSB conforming implementation shall provide the generic methods for Class `std::underflow_error` specified in Table 9-101, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-101 `libstdcxx` - Class `underflow_error` Function Interfaces**

<code>underflow_error::underflow_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>underflow_error::underflow_error(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>underflow_error::~~underflow_error()(GLIBCXX_3.4) [ISOCXX]</code>
<code>underflow_error::~~underflow_error()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::underflow_error` specified in Table 9-102, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-102 `libstdcxx` - Class `underflow_error` Data Interfaces**

<code>typeid for underflow_error(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for underflow_error(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for underflow_error(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.31 Class `invalid_argument`

#### 9.1.31.1 Class data for `invalid_argument`

The virtual table for the `std::invalid_argument` class is described by Table 9-103

**Table 9-103 Primary vtable for `invalid_argument`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid for invalid_argument</code>
<code>vfunc[0]:</code>	<code>invalid_argument::~~invalid_argument()</code>
<code>vfunc[1]:</code>	<code>invalid_argument::~~invalid_argument()</code>
<code>vfunc[2]:</code>	<code>logic_error::what() const</code>

The Run Time Type Information for the `std::invalid_argument` class is described by Table 9-104

**Table 9-104 `typeid` for `invalid_argument`**

Base Vtable	<code>vtable for __cxxabiv1::__si_class_type_info</code>
Name	<code>typeid name for invalid_argument</code>

### 9.1.31.2 Interfaces for Class `invalid_argument`

An LSB conforming implementation shall provide the generic methods for Class `std::invalid_argument` specified in Table 9-105, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-105 libstdcxx - Class `invalid_argument` Function Interfaces**

<code>invalid_argument::invalid_argument(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>invalid_argument::invalid_argument(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>invalid_argument::~~invalid_argument()(GLIBCXX_3.4) [ISOCXX]</code>
<code>invalid_argument::~~invalid_argument()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::invalid_argument` specified in Table 9-106, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-106 libstdcxx - Class `invalid_argument` Data Interfaces**

<code>typeid for invalid_argument(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for invalid_argument(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for invalid_argument(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.32 Class `bad_cast`

#### 9.1.32.1 Class data for `bad_cast`

The virtual table for the `std::bad_cast` class is described by Table 9-107

**Table 9-107 Primary vtable for `bad_cast`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid for bad_cast</code>
<code>vfunc[0]:</code>	<code>bad_cast::~~bad_cast()</code>
<code>vfunc[1]:</code>	<code>bad_cast::~~bad_cast()</code>
<code>vfunc[2]:</code>	<code>exception::what() const</code>

The Run Time Type Information for the `std::bad_cast` class is described by Table 9-108

**Table 9-108 `typeid` for `bad_cast`**

Base Vtable	<code>vtable for __cxxabiv1::__si_class_type_info</code>
Name	<code>typeid name for bad_cast</code>

### 9.1.32.2 Interfaces for Class `bad_cast`

An LSB conforming implementation shall provide the generic methods for Class `std::bad_cast` specified in Table 9-109, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-109 `libstdcxx` - Class `bad_cast` Function Interfaces**

<code>bad_cast::~bad_cast()</code> (GLIBCXX_3.4) [ISOCXX]
<code>bad_cast::~bad_cast()</code> (GLIBCXX_3.4) [ISOCXX]
<code>bad_cast::~bad_cast()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::bad_cast` specified in Table 9-110, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-110 `libstdcxx` - Class `bad_cast` Data Interfaces**

<code>typeid</code> for <code>bad_cast</code> (GLIBCXX_3.4) [CXXABI]
<code>typeid</code> name for <code>bad_cast</code> (GLIBCXX_3.4) [CXXABI]
<code>vtable</code> for <code>bad_cast</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.33 Class `bad_alloc`

#### 9.1.33.1 Class data for `bad_alloc`

The virtual table for the `std::bad_alloc` class is described by Table 9-111

**Table 9-111 Primary `vtable` for `bad_alloc`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid</code> for <code>bad_alloc</code>
<code>vfunc</code> [0]:	<code>bad_alloc::~bad_alloc()</code>
<code>vfunc</code> [1]:	<code>bad_alloc::~bad_alloc()</code>
<code>vfunc</code> [2]:	<code>exception::what() const</code>

The Run Time Type Information for the `std::bad_alloc` class is described by Table 9-112

**Table 9-112 `typeid` for `bad_alloc`**

Base <code>Vtable</code>	<code>vtable</code> for <code>__cxxabiv1::__si_class_type_info</code>
Name	<code>typeid</code> name for <code>bad_alloc</code>

#### 9.1.33.2 Interfaces for Class `bad_alloc`

An LSB conforming implementation shall provide the generic methods for Class `std::bad_alloc` specified in Table 9-113, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-113 libstdcxx - Class bad\_alloc Function Interfaces**

bad_alloc::~bad_alloc()(GLIBCXX_3.4) [ISOCXX]
bad_alloc::~bad_alloc()(GLIBCXX_3.4) [ISOCXX]
bad_alloc::~bad_alloc()(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class std::bad\_alloc specified in Table 9-114, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-114 libstdcxx - Class bad\_alloc Data Interfaces**

typeid for bad_alloc(GLIBCXX_3.4) [CXXABI]
typeid name for bad_alloc(GLIBCXX_3.4) [CXXABI]
vtable for bad_alloc(GLIBCXX_3.4) [CXXABI]

### 9.1.34 struct \_\_numeric\_limits\_base

#### 9.1.34.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct \_\_numeric\_limits\_base specified in Table 9-115, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-115 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]
__numeric_limits_base::radix(GLIBCXX_3.4) [ISOCXX]

<code>__numeric_limits_base::traps(GLIBCXX_3.4) [ISOCXX]</code>
<code>__numeric_limits_base::digits(GLIBCXX_3.4) [ISOCXX]</code>
<code>__numeric_limits_base::digits10(GLIBCXX_3.4) [ISOCXX]</code>
<code>__numeric_limits_base::is_exact(GLIBCXX_3.4) [ISOCXX]</code>
<code>__numeric_limits_base::is_iec559(GLIBCXX_3.4) [ISOCXX]</code>
<code>__numeric_limits_base::is_modulo(GLIBCXX_3.4) [ISOCXX]</code>
<code>__numeric_limits_base::is_signed(GLIBCXX_3.4) [ISOCXX]</code>

### 9.1.35 struct `numeric_limits<long double>`

#### 9.1.35.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for `struct numeric_limits<long double>` specified in Table 9-116, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-116 `libstdcxx - struct numeric_limits<long double>` Data Interfaces**

<code>numeric_limits&lt;long double&gt;::has_denorm(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::is_bounded(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::is_integer(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::round_style(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::has_infinity(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::max_exponent(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::min_exponent(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::is_specialized(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::max_exponent10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::min_exponent10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::tinyness_before(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::radix(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::traps(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::digits(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::digits10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::is_exact(GLIBCXX_3.4) [ISOCXX]</code>

<code>numeric_limits&lt;long double&gt;::is_iec559(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::is_modulo(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long double&gt;::is_signed(GLIBCXX_3.4) [ISOCXX]</code>

### 9.1.36 struct `numeric_limits<long long>`

#### 9.1.36.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct `numeric_limits<long long>` specified in Table 9-117, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-117 `libstdcxx` - struct `numeric_limits<long long>` Data Interfaces**

<code>numeric_limits&lt;long long&gt;::has_denorm(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::is_bounded(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::is_integer(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::round_style(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::has_infinity(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::max_exponent(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::min_exponent(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::is_specialized(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::max_exponent10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::min_exponent10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::tinyness_before(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::radix(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::traps(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::digits(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::digits10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::is_exact(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::is_iec559(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::is_modulo(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;long long&gt;::is_signed(GLIBCXX_3.4) [ISOCXX]</code>

## 9.1.37 struct numeric\_limits<unsigned long long>

### 9.1.37.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric\_limits<unsigned long long> specified in Table 9-118, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-118 libstdcxx - struct numeric\_limits<unsigned long long> Data Interfaces**

numeric_limits<unsigned long long>::has_denorm(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::is_bounded(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::is_integer(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::round_style(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::has_infinity(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::max_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::min_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::is_specialized(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::max_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::min_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::tinyness_before(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::radix(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::traps(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::digits(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::digits10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::is_exact(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long long>::is_iec559(GLIBCXX_3.4) [ISOCXX]



numeric_limits<unsigned long long>::is_modulo(GLIBCXX_3.4) [ISOCXX]
---

numeric_limits<unsigned long long>::is_signed(GLIBCXX_3.4) [ISOCXX]
---

## 9.1.38 struct numeric\_limits<float>

### 9.1.38.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric\_limits<float> specified in Table 9-119, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-119 libstdcxx - struct numeric\_limits<float> Data Interfaces**

numeric_limits<float>::has_denorm(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::is_bounded(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::is_integer(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::round_style(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::has_infinity(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::max_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::min_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::is_specialized(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::max_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::min_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::tinyness_before(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::radix(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::traps(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::digits(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::digits10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::is_exact(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::is_iec559(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::is_modulo(GLIBCXX_3.4) [ISOCXX]
numeric_limits<float>::is_signed(GLIBCXX_3.4) [ISOCXX]

## 9.1.39 struct numeric\_limits<double>

### 9.1.39.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric\_limits<double> specified in Table 9-120, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-120 libstdcxx - struct numeric\_limits<double> Data Interfaces**

numeric_limits<double>::has_denorm(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::is_bounded(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::is_integer(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::round_style(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::has_infinity(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::max_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::min_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::is_specialized(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::max_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::min_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::tinyness_before(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::radix(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::traps(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::digits(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::digits10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::is_exact(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::is_iec559(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::is_modulo(GLIBCXX_3.4) [ISOCXX]
numeric_limits<double>::is_signed(GLIBCXX_3.4) [ISOCXX]

## 9.1.40 struct numeric\_limits<short>

### 9.1.40.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct `numeric_limits<short>` specified in Table 9-121, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-121 libstdcxx - struct `numeric_limits<short>` Data Interfaces**

<code>numeric_limits&lt;short&gt;::has_denorm(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::is_bounded(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::is_integer(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::round_style(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::has_infinity(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::max_exponent(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::min_exponent(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::is_specialized(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::max_exponent10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::min_exponent10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::tinyness_before(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::radix(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::traps(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::digits(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::digits10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::is_exact(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::is_iec559(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::is_modulo(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;short&gt;::is_signed(GLIBCXX_3.4) [ISOCXX]</code>

## 9.1.41 struct `numeric_limits<unsigned short>`

### 9.1.41.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct `numeric_limits<unsigned short>` specified in Table 9-122, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-122 libstdcxx - struct `numeric_limits<unsigned short>` Data Interfaces**

<code>numeric_limits&lt;unsigned short&gt;::has_denorm(GLIBCXX_3.4) [ISOCXX]</code>
---

numeric_limits<unsigned short>::is_bounded(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::is_integer(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::round_style(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::has_infinity(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::max_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::min_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::is_specialized(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::max_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::min_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::tinyness_before(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::radix(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::traps(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::digits(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::digits10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::is_exact(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::is_iec559(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::is_modulo(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned short>::is_signed(GLIBCXX_3.4) [ISOCXX]

## 9.1.42 struct numeric\_limits<int>

### 9.1.42.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric\_limits<int> specified in Table 9-123, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-123 libstdcxx - struct numeric\_limits<int> Data Interfaces**

numeric_limits<int>::has_denorm(GLIBCXX_3.4) [ISOCXX]
numeric_limits<int>::is_bounded(GLIBCXX_3.4) [ISOCXX]
numeric_limits<int>::is_integer(GLIBCXX_3.4) [ISOCXX]
numeric_limits<int>::round_style(GLIBCXX_3.4) [ISOCXX]
numeric_limits<int>::has_infinity(GLIBCXX_3.4) [ISOCXX]

<code>numeric_limits&lt;int&gt;::max_exponent</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::min_exponent</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::has_quiet_NaN</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::is_specialized</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::max_exponent10</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::min_exponent10</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::has_denorm_loss</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::tinyness_before</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::has_signaling_NaN</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::radix</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::traps</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::digits</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::digits10</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::is_exact</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::is_iec559</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::is_modulo</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;int&gt;::is_signed</code> (GLIBCXX_3.4) [ISOCXX]

### 9.1.43 struct `numeric_limits<unsigned int>`

#### 9.1.43.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct `numeric_limits<unsigned int>` specified in Table 9-124, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-124 libstdcxx - struct `numeric_limits<unsigned int>` Data Interfaces**

<code>numeric_limits&lt;unsigned int&gt;::has_denorm</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;unsigned int&gt;::is_bounded</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;unsigned int&gt;::is_integer</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;unsigned int&gt;::round_style</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;unsigned int&gt;::has_infinity</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;unsigned int&gt;::max_exponent</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;unsigned int&gt;::min_exponent</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;unsigned int&gt;::has_quiet_NaN</code> (GLIBCXX_3.4) [ISOCXX]
<code>numeric_limits&lt;unsigned int&gt;::is_specialized</code> (GLIBCXX_3.4) [ISOCXX]

numeric_limits<unsigned int>::max_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::min_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::tinyness_before(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::radix(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::traps(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::digits(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::digits10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::is_exact(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::is_iec559(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::is_modulo(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned int>::is_signed(GLIBCXX_3.4) [ISOCXX]

### 9.1.44 struct numeric\_limits<long>

#### 9.1.44.1 Interfaces for struct numeric\_limits<long>

No external methods are defined for libstdc++ - struct numeric\_limits<long> in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric\_limits<long> specified in Table 9-125, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-125 libstdc++ - struct numeric\_limits<long> Data Interfaces**

numeric_limits<long>::has_denorm(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::is_bounded(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::is_integer(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::round_style(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::has_infinity(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::max_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::min_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::is_specialized(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::max_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::min_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::tinyness_before(GLIBCXX_3.4) [ISOCXX]

numeric_limits<long>::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::radix(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::traps(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::digits(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::digits10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::is_exact(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::is_iec559(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::is_modulo(GLIBCXX_3.4) [ISOCXX]
numeric_limits<long>::is_signed(GLIBCXX_3.4) [ISOCXX]

### 9.1.45 struct numeric\_limits<unsigned long>

#### 9.1.45.1 Interfaces for struct numeric\_limits<unsigned long>

No external methods are defined for libstdc++ - struct numeric\_limits<unsigned long> in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric\_limits<unsigned long> specified in Table 9-126, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-126 libstdc++ - struct numeric\_limits<unsigned long> Data Interfaces**

numeric_limits<unsigned long>::has_denorm(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::is_bounded(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::is_integer(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::round_style(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::has_infinity(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::max_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::min_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::is_specialized(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::max_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::min_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::tinyness_before(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::radix(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::traps(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::digits(GLIBCXX_3.4) [ISOCXX]

numeric_limits<unsigned long>::digits10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::is_exact(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::is_iec559(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::is_modulo(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned long>::is_signed(GLIBCXX_3.4) [ISOCXX]

## 9.1.46 struct numeric\_limits<wchar\_t>

### 9.1.46.1 Interfaces for struct numeric\_limits<wchar\_t>

No external methods are defined for libstdc++ - struct numeric\_limits<wchar\_t> in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric\_limits<wchar\_t> specified in Table 9-127, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-127 libstdc++ - struct numeric\_limits<wchar\_t> Data Interfaces**

numeric_limits<wchar_t>::has_denorm(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::is_bounded(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::is_integer(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::round_style(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::has_infinity(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::max_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::min_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::is_specialized(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::max_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::min_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::tinyness_before(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::radix(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::traps(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::digits(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::digits10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::is_exact(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::is_iec559(GLIBCXX_3.4) [ISOCXX]
numeric_limits<wchar_t>::is_modulo(GLIBCXX_3.4) [ISOCXX]



numeric_limits<wchar_t>::is_signed(GLIBCXX_3.4) [ISOCXX]
--

## 9.1.47 struct numeric\_limits<unsigned char>

### 9.1.47.1 Interfaces for struct numeric\_limits<unsigned char>

No external methods are defined for libstdc++ - struct numeric\_limits<unsigned char> in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric\_limits<unsigned char> specified in Table 9-128, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-128 libstdc++ - struct numeric\_limits<unsigned char> Data Interfaces**

numeric_limits<unsigned char>::has_denorm(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::is_bounded(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::is_integer(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::round_style(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::has_infinity(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::max_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::min_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::is_specialized(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::max_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::min_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::tinyness_before(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::radix(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::traps(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::digits(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::digits10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::is_exact(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::is_iec559(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::is_modulo(GLIBCXX_3.4) [ISOCXX]
numeric_limits<unsigned char>::is_signed(GLIBCXX_3.4) [ISOCXX]

## 9.1.48 struct numeric\_limits<signed char>

### 9.1.48.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct numeric\_limits<signed char> specified in Table 9-129, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-129 libstdcxx - struct numeric\_limits<signed char> Data Interfaces**

numeric_limits<signed char>::has_denorm(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::is_bounded(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::is_integer(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::round_style(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::has_infinity(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::max_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::min_exponent(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::is_specialized(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::max_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::min_exponent10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::tinyness_before(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::radix(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::traps(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::digits(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::digits10(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::is_exact(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::is_iec559(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::is_modulo(GLIBCXX_3.4) [ISOCXX]
numeric_limits<signed char>::is_signed(GLIBCXX_3.4) [ISOCXX]

## 9.1.49 struct numeric\_limits<char>

### 9.1.49.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct `numeric_limits<char>` specified in Table 9-130, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-130 libstdcxx - struct `numeric_limits<char>` Data Interfaces**

<code>numeric_limits&lt;char&gt;::has_denorm(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::is_bounded(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::is_integer(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::round_style(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::has_infinity(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::max_exponent(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::min_exponent(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::is_specialized(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::max_exponent10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::min_exponent10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::tinyness_before(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::radix(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::traps(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::digits(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::digits10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::is_exact(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::is_iec559(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::is_modulo(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;char&gt;::is_signed(GLIBCXX_3.4) [ISOCXX]</code>

## 9.1.50 struct `numeric_limits<bool>`

### 9.1.50.1 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for struct `numeric_limits<bool>` specified in Table 9-131, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-131 libstdcxx - struct `numeric_limits<bool>` Data Interfaces**

<code>numeric_limits&lt;bool&gt;::has_denorm(GLIBCXX_3.4) [ISOCXX]</code>
---

<code>numeric_limits&lt;bool&gt;::is_bounded(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::is_integer(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::round_style(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::has_infinity(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::max_exponent(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::min_exponent(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::has_quiet_NaN(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::is_specialized(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::max_exponent10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::min_exponent10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::has_denorm_loss(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::tinyness_before(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::has_signaling_NaN(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::radix(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::traps(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::digits(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::digits10(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::is_exact(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::is_iec559(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::is_modulo(GLIBCXX_3.4) [ISOCXX]</code>
<code>numeric_limits&lt;bool&gt;::is_signed(GLIBCXX_3.4) [ISOCXX]</code>

## 9.1.51 Class `ctype_base`

### 9.1.51.1 Class data for `ctype_base`

The Run Time Type Information for the `std::ctype_base` class is described by Table 9-132

**Table 9-132** `typeinfo` for `ctype_base`

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

### 9.1.51.2 Interfaces for Class `ctype_base`

No external methods are defined for `libstdc++` - Class `std::ctype_base` in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for Class `std::ctype_base` specified in Table 9-133, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-133 libstdcxx - Class ctype\_base Data Interfaces**

ctype_base::alnum(GLIBCXX_3.4) [ISOCXX]
ctype_base::alpha(GLIBCXX_3.4) [ISOCXX]
ctype_base::cntrl(GLIBCXX_3.4) [ISOCXX]
ctype_base::digit(GLIBCXX_3.4) [ISOCXX]
ctype_base::graph(GLIBCXX_3.4) [ISOCXX]
ctype_base::lower(GLIBCXX_3.4) [ISOCXX]
ctype_base::print(GLIBCXX_3.4) [ISOCXX]
ctype_base::punct(GLIBCXX_3.4) [ISOCXX]
ctype_base::space(GLIBCXX_3.4) [ISOCXX]
ctype_base::upper(GLIBCXX_3.4) [ISOCXX]
ctype_base::xdigit(GLIBCXX_3.4) [ISOCXX]
typeid for ctype_base(GLIBCXX_3.4) [CXXABI]
typeid name for ctype_base(GLIBCXX_3.4) [CXXABI]

**9.1.52 Class `__ctype_abstract_base<char>`****9.1.52.1 Class data for `__ctype_abstract_base<char>`**

The virtual table for the `std::__ctype_abstract_base<char>` class is described by Table 9-134

**Table 9-134 Primary vtable for `__ctype_abstract_base<char>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>__ctype_abstract_base&lt;char&gt;</code>
vfunc[0]:	
vfunc[1]:	
vfunc[2]:	<code>__cxa_pure_virtual</code>
vfunc[3]:	<code>__cxa_pure_virtual</code>
vfunc[4]:	<code>__cxa_pure_virtual</code>
vfunc[5]:	<code>__cxa_pure_virtual</code>
vfunc[6]:	<code>__cxa_pure_virtual</code>
vfunc[7]:	<code>__cxa_pure_virtual</code>
vfunc[8]:	<code>__cxa_pure_virtual</code>
vfunc[9]:	<code>__cxa_pure_virtual</code>
vfunc[10]:	<code>__cxa_pure_virtual</code>
vfunc[11]:	<code>__cxa_pure_virtual</code>

vfunc[12]:	__cxa_pure_virtual
vfunc[13]:	__cxa_pure_virtual

### 9.1.52.2 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for Class `std::__ctype_abstract_base<char>` specified in Table 9-135, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-135 `libstdcxx` - Class `__ctype_abstract_base<char>` Data Interfaces**

typeid for <code>__ctype_abstract_base&lt;char&gt;</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>__ctype_abstract_base&lt;char&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>__ctype_abstract_base&lt;char&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.53 Class `__ctype_abstract_base<wchar_t>`

#### 9.1.53.1 Class data for `__ctype_abstract_base<wchar_t>`

The virtual table for the `std::__ctype_abstract_base<wchar_t>` class is described by Table 9-136

**Table 9-136 Primary vtable for `__ctype_abstract_base<wchar_t>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>__ctype_abstract_base&lt;wchar_t&gt;</code>
vfunc[0]:	
vfunc[1]:	
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
vfunc[9]:	__cxa_pure_virtual
vfunc[10]:	__cxa_pure_virtual
vfunc[11]:	__cxa_pure_virtual
vfunc[12]:	__cxa_pure_virtual

vfunc[13]:	__cxa_pure_virtual
------------	--------------------

### 9.1.53.2 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for Class `std::__ctype_abstract_base<wchar_t>` specified in Table 9-137, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-137 libstdcxx - Class `__ctype_abstract_base<wchar_t>` Data Interfaces**

<code>typeid</code> for <code>__ctype_abstract_base&lt;wchar_t&gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>typeid</code> name for <code>__ctype_abstract_base&lt;wchar_t&gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>vtable</code> for <code>__ctype_abstract_base&lt;wchar_t&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.54 Class `ctype<char>`

#### 9.1.54.1 Class data for `ctype<char>`

The virtual table for the `std::ctype<char>` class is described by Table 9-138

**Table 9-138 Primary vtable for `ctype<char>`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid</code> for <code>ctype&lt;char&gt;</code>
<code>vfunc</code> [0]:	<code>ctype&lt;char&gt;::~~ctype()</code>
<code>vfunc</code> [1]:	<code>ctype&lt;char&gt;::~~ctype()</code>
<code>vfunc</code> [2]:	<code>ctype&lt;char&gt;::do_toupper(char) const</code>
<code>vfunc</code> [3]:	<code>ctype&lt;char&gt;::do_toupper(char*, char const*) const</code>
<code>vfunc</code> [4]:	<code>ctype&lt;char&gt;::do_tolower(char) const</code>
<code>vfunc</code> [5]:	<code>ctype&lt;char&gt;::do_tolower(char*, char const*) const</code>
<code>vfunc</code> [6]:	<code>ctype&lt;char&gt;::do_widen(char) const</code>
<code>vfunc</code> [7]:	<code>ctype&lt;char&gt;::do_widen(char const*, char const*, char*) const</code>
<code>vfunc</code> [8]:	<code>ctype&lt;char&gt;::do_narrow(char, char) const</code>
<code>vfunc</code> [9]:	<code>ctype&lt;char&gt;::do_narrow(char const*, char const*, char, char*) const</code>

### 9.1.54.2 Interfaces for Class `ctype<char>`

An LSB conforming implementation shall provide the generic methods for Class `std::ctype<char>` specified in Table 9-139, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-139 libstdcxx - Class `ctype<char>` Function Interfaces**

<code>ctype&lt;char&gt;::do_tolower(char*, char const*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::do_tolower(char) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::do_toupper(char*, char const*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::do_toupper(char) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::do_widen(char const*, char const*, char*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::do_widen(char) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::do_narrow(char const*, char const*, char, char*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::do_narrow(char, char) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::classic_table()</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::~~ctype()</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::~~ctype()</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::~~ctype()</code> (GLIBCXX_3.4) [ISOCXX]
<code>bool has_facet&lt;ctype&lt;char&gt;&gt;(locale const&amp;)</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::ctype<char>` specified in Table 9-140, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-140 libstdcxx - Class `ctype<char>` Data Interfaces**

<code>ctype&lt;char&gt;::table_size</code> (GLIBCXX_3.4) [ISOCXX]
<code>ctype&lt;char&gt;::id</code> (GLIBCXX_3.4) [ISOCXX]
<code>typeid for ctype&lt;char&gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>typeid name for ctype&lt;char&gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>vtable for ctype&lt;char&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.55 Class `ctype<wchar_t>`

#### 9.1.55.1 Class data for `ctype<wchar_t>`

The virtual table for the `std::ctype<wchar_t>` class is described by Table 9-141

**Table 9-141 Primary vtable for `ctype<wchar_t>`**

Base Offset	0
Virtual Base Offset	0



RTTI	typeinfo for ctype<wchar_t>
vfunc[0]:	ctype<wchar_t>::~~ctype()
vfunc[1]:	ctype<wchar_t>::~~ctype()
vfunc[2]:	ctype<wchar_t>::do_is(unsigned short, wchar_t) const
vfunc[3]:	ctype<wchar_t>::do_is(wchar_t const*, wchar_t const*, unsigned short*) const
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 9-142

**Table 9-142 typeinfo for ctype<wchar\_t>**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for ctype<wchar_t>

### 9.1.55.2 Interfaces for Class ctype<wchar\_t>

An LSB conforming implementation shall provide the generic methods for Class std::ctype<wchar\_t> specified in Table 9-143, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-143 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_tolower(wchar_t) const(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::do_toupper(wchar_t*, wchar_t const*) const(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::do_toupper(wchar_t) 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>::M_convert_to_wmask(unsigned short) const(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::do_is(wchar_t const*, wchar_t const*, unsigned short*) const(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::do_is(unsigned short, wchar_t) const(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::do_widen(char const*, char const*, wchar_t*) const(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::do_widen(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_narrow(wchar_t, char) const(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::M_initialize_ctype()(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::~ctype()(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::~ctype()(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::~ctype()(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class std::ctype<wchar\_t> specified in Table 9-144, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-144 libstdcxx - Class ctype<wchar\_t> Data Interfaces**

ctype<wchar_t>::id(GLIBCXX_3.4) [ISOCXX]
typeid for ctype<wchar_t>(GLIBCXX_3.4) [CXXABI]
typeid name for ctype<wchar_t>(GLIBCXX_3.4) [CXXABI]
vtable for ctype<wchar_t>(GLIBCXX_3.4) [CXXABI]

## 9.1.56 Class `ctype_byname<char>`

### 9.1.56.1 Class data for `ctype_byname<char>`

The virtual table for the `std::ctype_byname<char>` class is described by Table 9-145

**Table 9-145 Primary vtable for `ctype_byname<char>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>ctype_byname&lt;char&gt;</code>
<code>vfunc[0]:</code>	<code>ctype_byname&lt;char&gt;::~~ctype_byname()</code>
<code>vfunc[1]:</code>	<code>ctype_byname&lt;char&gt;::~~ctype_byname()</code>
<code>vfunc[2]:</code>	<code>ctype&lt;char&gt;::do_toupper(char) const</code>
<code>vfunc[3]:</code>	<code>ctype&lt;char&gt;::do_toupper(char*, char const*) const</code>
<code>vfunc[4]:</code>	<code>ctype&lt;char&gt;::do_tolower(char) const</code>
<code>vfunc[5]:</code>	<code>ctype&lt;char&gt;::do_tolower(char*, char const*) const</code>
<code>vfunc[6]:</code>	<code>ctype&lt;char&gt;::do_widen(char) const</code>
<code>vfunc[7]:</code>	<code>ctype&lt;char&gt;::do_widen(char const*, char const*, char*) const</code>
<code>vfunc[8]:</code>	<code>ctype&lt;char&gt;::do_narrow(char, char) const</code>
<code>vfunc[9]:</code>	<code>ctype&lt;char&gt;::do_narrow(char const*, char const*, char, char*) const</code>

The Run Time Type Information for the `std::ctype_byname<char>` class is described by Table 9-146

**Table 9-146 typeinfo for `ctype_byname<char>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>ctype_byname&lt;char&gt;</code>

### 9.1.56.2 Interfaces for Class `ctype_byname<char>`

An LSB conforming implementation shall provide the generic methods for Class `std::ctype_byname<char>` specified in Table 9-147, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-147 libstdcxx - Class `ctype_byname<char>` Function Interfaces**

<code>ctype_byname&lt;char&gt;::~~ctype_byname()(GLIBCXX_3.4) [ISOCXX]</code>
---

<code>ctype_byname&lt;char&gt;::~~ctype_byname()(GLIBCXX_3.4) [ISOCXX]</code>
---

<code>ctype_byname&lt;char&gt;::~~ctype_byname()(GLIBCXX_3.4) [ISOCXX]</code>
---

An LSB conforming implementation shall provide the generic data interfaces for Class `std::ctype_byname<char>` specified in Table 9-148, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-148 libstdcxx - Class `ctype_byname<char>` Data Interfaces**

<code>typeid for ctype_byname&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>
--

<code>typeid name for ctype_byname&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>
---

<code>vtable for ctype_byname&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>
--

### 9.1.57 Class `ctype_byname<wchar_t>`

#### 9.1.57.1 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 9-149, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-149 libstdcxx - Class `ctype_byname<wchar_t>` Function Interfaces**

<code>ctype_byname&lt;wchar_t&gt;::~~ctype_byname()(GLIBCXX_3.4) [CXXABI]</code>
--

<code>ctype_byname&lt;wchar_t&gt;::~~ctype_byname()(GLIBCXX_3.4) [CXXABI]</code>
--

<code>ctype_byname&lt;wchar_t&gt;::~~ctype_byname()(GLIBCXX_3.4) [CXXABI]</code>
--

An LSB conforming implementation shall provide the generic data interfaces for Class `std::ctype_byname<wchar_t>` specified in Table 9-150, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-150 libstdcxx - Class `ctype_byname<wchar_t>` Data Interfaces**

<code>typeid for ctype_byname&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
---

<code>typeid name for ctype_byname&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
--

<code>vtable for ctype_byname&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
---

### 9.1.58 Class `basic_string<char, char_traits<char>, allocator<char>>`

#### 9.1.58.1 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 9-151, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-151 libstdcxx - Class `basic_string<char, char_traits<char>, allocator<char>>` Function Interfaces**

<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::get_allocator()</code>
--

const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::end() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::Rep::_M_is_leaked() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::Rep::_M_is_shared() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::data() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::rend() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::size() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::begin() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::c_str() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::empty() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_rep() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::length() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::rbegin() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_data() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_iend() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::compare(char const*) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::compare(basic_string<char, char_traits<char>, allocator<char> > const&) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::capacity() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::max_size() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_ibegin() const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_Alloc_hider::_Alloc_hider(char*, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]

<pre>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) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_M_leak_hard()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_S_empty_rep()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_S_copy_chars(char*, __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) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_S_copy_chars(char*, __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*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&gt;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_S_copy_chars(char*, char const*, char const*)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_S_copy_chars(char*, char*, char*)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::end()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Rep::_M_destroy(allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Rep::_M_dispose(allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Rep::_M_refcopy()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Rep::_M_refdata()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Rep::_S_empty_rep()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Rep::_M_set_leaked()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Rep::_M_set_sharable()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_Rep::_M_grab(allocator&lt;char&gt; const&amp;, allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_rend()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::_swap(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>

<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::begin()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::clear()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::erase(__gnu_cxx::__normal_iterator&lt;char*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&gt;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::erase(__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*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&gt;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::append(char const*)</code> (GLIBCXX_3.4) [ISOCXX]
<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;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::assign(char const*)</code> (GLIBCXX_3.4) [ISOCXX]
<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;)</code> (GLIBCXX_3.4) [ISOCXX]
<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)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::rbegin()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::_M_data(char*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::_M_leak()</code> (GLIBCXX_3.4) [ISOCXX]
<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*, 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;, __gnu_cxx::__normal_iterator&lt;char const*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&gt;)</code> (GLIBCXX_3.4) [ISOCXX]
<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*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&gt;, char const*)</code> (GLIBCXX_3.4) [ISOCXX]
<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*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&gt;, char const*)</code> (GLIBCXX_3.4) [ISOCXX]

<pre>char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&gt;, __gnu_cxx::__normal_iterator&lt;char*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&gt;, char const*, char const*)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>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*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&gt;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>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*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&gt;, char*, char*)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>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*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&gt;, __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*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&gt;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::push_back(char)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_string(char const*, allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_string(allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_string(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_string()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_string&lt;__gnu_cxx::__normal_iterator&lt;char*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&gt; &gt;(__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*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&gt;, allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_string&lt;char const*&gt;(char const*, char const*, allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_string&lt;char*&gt;(char*, char*, allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_string(char const*, allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;</pre>



<code>&gt;::basic_string(allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::basic_string(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::basic_string()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::basic_string&lt;__gnu_cxx::__normal_iterator&lt;char*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&gt;&gt; &gt;(__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*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&gt;, allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::basic_string&lt;char const*&gt;(char const*, char const*, allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::basic_string&lt;char*&gt;(char*, char*, allocator&lt;char&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::~basic_string()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::~basic_string()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::operator=(char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::operator=(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::operator=(char)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::operator+=(char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::operator+=(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::operator+=(char)(GLIBCXX_3.4) [ISOCXX]</code>

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 9-152, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-152 libstdcxx - Class `basic_string<char, char_traits<char>, allocator<char>>` Data Interfaces**

<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;</code>
---

<code>&gt;::_Rep::_S_max_size(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::_Rep::_S_terminal(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;::_Rep::_S_empty_rep_storage(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::_npos(GLIBCXX_3.4) [ISOCXX]</code>

### 9.1.59 Class `basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

#### 9.1.59.1 Interfaces for Class `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, std::char_traits<wchar_t>, std::allocator<wchar_t>>` specified in Table 9-153, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-153 libstdcxx - Class `basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>` Function Interfaces**

<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt;::get_allocator() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::end() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt;::_Rep::_M_is_leaked() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt;::_Rep::_M_is_shared() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::data() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::rend() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::size() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::begin() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::c_str() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::empty() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_M_rep() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::length() const(GLIBCXX_3.4) [ISOCXX]</code>

<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::rbegin()</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::_M_data() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::_M_iend() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::compare(wchar_t const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::compare(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt; const&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::capacity()</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::max_size() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::_M_ibegin() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::_Alloc_hider::_Alloc_hider(wchar_t*, allocator&lt;wchar_t&gt;</code> <code>const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::_Alloc_hider::_Alloc_hider(wchar_t*, allocator&lt;wchar_t&gt;</code> <code>const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::_M_leak_hard()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::_S_empty_rep()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::_S_copy_chars(wchar_t*, __gnu_cxx::__normal_iterator&lt;wchar_t const*</code> <code>, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt;,</code> <code>__gnu_cxx::__normal_iterator&lt;wchar_t const*, basic_string&lt;wchar_t,</code> <code>char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt; &gt;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::_S_copy_chars(wchar_t*, __gnu_cxx::__normal_iterator&lt;wchar_t*</code> <code>, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt;,</code> <code>__gnu_cxx::__normal_iterator&lt;wchar_t*, basic_string&lt;wchar_t,</code> <code>char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; &gt; &gt;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::_S_copy_chars(wchar_t*, wchar_t const*, wchar_t const*)(GLIBCXX_3.4)</code> <code>[ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::_S_copy_chars(wchar_t*, wchar_t*, wchar_t*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> <code>&gt;::end()(GLIBCXX_3.4) [ISOCXX]</code>

<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_Rep::_M_destroy(allocator&lt;wchar_t&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_Rep::_M_dispose(allocator&lt;wchar_t&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_Rep::_M_refcopy()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_Rep::_M_refdata()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_Rep::_S_empty_rep()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_Rep::_M_set_leaked()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_Rep::_M_set_sharable()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_Rep::_M_grab(allocator&lt;wchar_t&gt; const&amp;, allocator&lt;wchar_t&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_rend()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::swap(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_begin()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_clear()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_erase(__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) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_erase(__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) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_append(wchar_t const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_append(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_assign(wchar_t const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_assign(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>





<code>const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::basic_string&lt;wchar_t const*&gt;(wchar_t const*, wchar_t const*, allocator&lt;wchar_t&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::basic_string&lt;wchar_t*&gt;(wchar_t*, wchar_t*, allocator&lt;wchar_t&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::~~basic_string()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::~~basic_string()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::operator=(wchar_t const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::operator=(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::operator=(wchar_t)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::operator+=(wchar_t const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::operator+=(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::operator+=(wchar_t)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; operator+&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;(char const*, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; operator+&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; operator+&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;(char, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; operator+&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;(wchar_t const*, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; operator+&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; operator+&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code>

operator+<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>(wchar_t, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>> const&)(GLIBCXX_3.4) [ISOCXX]
---

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 9-154, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-154 libstdc++ - Class `basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>` Data Interfaces**

<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_Rep::_S_max_size</code> (GLIBCXX_3.4) [ISOCXX]
--

<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_Rep::_S_terminal</code> (GLIBCXX_3.4) [ISOCXX]
--

<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_Rep::_S_empty_rep_storage</code> (GLIBCXX_3.4) [ISOCXX]
---

<code>basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::_npos</code> (GLIBCXX_3.4) [ISOCXX]
--

### 9.1.60 Class `basic_stringstream<char, char_traits<char>, allocator<char>>`

#### 9.1.60.1 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 ISO/IEC 23360.

The VTT for the `std::basic_stringstream<char, std::char_traits<char>, std::allocator<char>>` class is described by Table 9-155

**Table 9-155 VTT for `basic_stringstream<char, char_traits<char>, allocator<char>>`**

VTT Name	<code>_ZTTSt18basic_stringstreamIcSt11char_traitsIcESaIcEE</code>
Number of Entries	10

#### 9.1.60.2 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 9-156, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-156 libstdc++ - Class `basic_stringstream<char, char_traits<char>, allocator<char>>` Function Interfaces**

<code>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::str()</code> <code>const</code> (GLIBCXX_3.4) [ISOCXX]
---



<code>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::rdbuf()</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>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]</code>
<code>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]</code>
<code>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_stringstream(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>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]</code>
<code>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::basic_stringstream(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::~~basic_stringstream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::~~basic_stringstream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::~~basic_stringstream()(GLIBCXX_3.4) [ISOCXX]</code>

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 9-157, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-157 libstdcxx - Class `basic_stringstream<char, char_traits<char>, allocator<char> >` Data Interfaces**

<code>typeid for basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>VTT for basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for basic_stringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.61 Class `basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >`

#### 9.1.61.1 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 ISO/IEC 23360.

The VTT for the `std::basic_stringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> >` class is described by Table 9-158

**Table 9-158 VTT for `basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >`**

VTT Name	<code>_ZTTSt18basic_stringstreamIwSt11char_traitsIwESaIwEE</code>
Number of Entries	10

### 9.1.61.2 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 9-159, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-159 `libstdcxx` - Class `basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >` Function Interfaces**

<code>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::str() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::rdbuf() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>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;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_stringstream(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_stringstream(_Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_stringstream(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_stringstream(_Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~~basic_stringstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~~basic_stringstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_stringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~~basic_stringstream()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_stringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> >` specified in Table 9-160, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-160 libstdcxx - Class basic\_stringstream<wchar\_t, char\_traits<wchar\_t>, allocator<wchar\_t>> Data Interfaces**

typeid for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>(GLIBCXX_3.4) [CXXABI]
typeid name for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>(GLIBCXX_3.4) [CXXABI]
VTT for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>(GLIBCXX_3.4) [CXXABI]
vtable for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>(GLIBCXX_3.4) [CXXABI]

### 9.1.62 Class basic\_istringstream<char, char\_traits<char>, allocator<char>>

#### 9.1.62.1 Class data for basic\_istringstream<char, char\_traits<char>, allocator<char>>

The virtual table for the std::basic\_istringstream<char, std::char\_traits<char>, std::allocator<char>> class is described in the relevant architecture specific part of ISO/IEC 23360.

The VTT for the std::basic\_istringstream<char, std::char\_traits<char>, std::allocator<char>> class is described by Table 9-161

**Table 9-161 VTT for basic\_istringstream<char, char\_traits<char>, allocator<char>>**

VTT Name	_ZTTSt19basic_istringstreamIcSt11char_traitsIcESaIcEE
Number of Entries	4

#### 9.1.62.2 Interfaces for Class basic\_istringstream<char, char\_traits<char>, allocator<char>>

An LSB conforming implementation shall provide the generic methods for Class std::basic\_istringstream<char, std::char\_traits<char>, std::allocator<char>> specified in Table 9-162, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-162 libstdcxx - Class basic\_istringstream<char, char\_traits<char>, allocator<char>> Function Interfaces**

basic_istringstream<char, char_traits<char>, allocator<char>>::str() const(GLIBCXX_3.4) [ISOCXX]
basic_istringstream<char, char_traits<char>, allocator<char>>::rdbuf() const(GLIBCXX_3.4) [ISOCXX]
basic_istringstream<char, char_traits<char>, allocator<char>>::str(basic_string<char, char_traits<char>, allocator<char>> const&)(GLIBCXX_3.4) [ISOCXX]
basic_istringstream<char, char_traits<char>, allocator<char>>::basic_istringstream(basic_string<char, char_traits<char>, allocator<char>> const&, _ios_Openmode)(GLIBCXX_3.4) [ISOCXX]

<code>basic_istream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::basic_istream(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::basic_istream(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::basic_istream(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::~~basic_istream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::~~basic_istream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::~~basic_istream()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_istream<char, std::char_traits<char>, std::allocator<char>>` specified in Table 9-163, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-163 libstdcxx - Class `basic_istream<char, char_traits<char>, allocator<char>>` Data Interfaces**

<code>typeid for basic_istream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for basic_istream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>VTT for basic_istream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for basic_istream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.63 Class `basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

#### 9.1.63.1 Class data for `basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

The virtual table for the `std::basic_istream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>>` class is described in the relevant architecture specific part of ISO/IEC 23360.

The VTT for the `std::basic_istream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>>` class is described by Table 9-164

**Table 9-164 VTT for `basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`**

VTT Name	<code>_ZTTSt19basic_istreamIwSt11char_traitsIwESaIwEE</code>
Number of Entries	4

### 9.1.63.2 Interfaces for Class `basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

An LSB conforming implementation shall provide the generic methods for Class `std::basic_istream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>>` specified in Table 9-165, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-165 libstdcxx - Class `basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>` Function Interfaces**

<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::str() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::rdbuf() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&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;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::basic_istream(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::basic_istream(_Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::basic_istream(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::basic_istream(_Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::~~basic_istream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::~~basic_istream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::~~basic_istream()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_istream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>>` specified in Table 9-166, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-166 libstdcxx - Class `basic_istream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>` Data Interfaces**

<code>typeid</code> for <code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>typeid</code> name for <code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
VTT for <code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]

vtable for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >(GLIBCXX_3.4) [CXXABI]
--

### 9.1.64 Class basic\_ostringstream<char, char\_traits<char>, allocator<char> >

#### 9.1.64.1 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 ISO/IEC 23360.

The VTT for the std::basic\_ostringstream<char, std::char\_traits<char>, std::allocator<char> > class is described by Table 9-167

**Table 9-167 VTT for basic\_ostringstream<char, char\_traits<char>, allocator<char> >**

VTT Name	_ZTTSt19basic_ostringstreamIcSt11char_traitsIcESaIcEE
Number of Entries	4

#### 9.1.64.2 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 9-168, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-168 libstdcxx - Class basic\_ostringstream<char, char\_traits<char>, allocator<char> > Function Interfaces**

basic_ostringstream<char, char_traits<char>, allocator<char> >::str() const(GLIBCXX_3.4) [ISOCXX]
basic_ostringstream<char, char_traits<char>, allocator<char> >::rdbuf() const(GLIBCXX_3.4) [ISOCXX]
basic_ostringstream<char, char_traits<char>, allocator<char> >::str(basic_string<char, char_traits<char>, allocator<char> > const&)(GLIBCXX_3.4) [ISOCXX]
basic_ostringstream<char, char_traits<char>, allocator<char> >::basic_ostringstream(basic_string<char, char_traits<char>, allocator<char> > const&, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_ostringstream<char, char_traits<char>, allocator<char> >::basic_ostringstream(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_ostringstream<char, char_traits<char>, allocator<char> >::basic_ostringstream(basic_string<char, char_traits<char>, allocator<char> > const&, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_ostringstream<char, char_traits<char>, allocator<char> >::basic_ostringstream(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_ostringstream<char, char_traits<char>, allocator<char> >

>::~~basic_ostringstream()(GLIBCXX_3.4) [ISOCXX]
basic_ostringstream<char, char_traits<char>, allocator<char>> >::~~basic_ostringstream()(GLIBCXX_3.4) [ISOCXX]
basic_ostringstream<char, char_traits<char>, allocator<char>> >::~~basic_ostringstream()(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_ostringstream<char, std::char_traits<char>, std::allocator<char>>` specified in Table 9-169, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-169 libstdcxx - Class `basic_ostringstream<char, char_traits<char>, allocator<char>>` Data Interfaces**

<code>typename basic_ostringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;</code> <code>&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typename basic_ostringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;</code> <code>&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>VTT for basic_ostringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;</code> <code>&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for basic_ostringstream&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;</code> <code>&gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.65 Class `basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

#### 9.1.65.1 Class data for `basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

The virtual table for the `std::basic_ostringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>>` class is described in the relevant architecture specific part of ISO/IEC 23360.

The VTT for the `std::basic_ostringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>>` class is described by Table 9-170

**Table 9-170 VTT for `basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`**

VTT Name	<code>_ZTTSt19basic_ostringstreamIwchar_traitsIwESaIwEE</code>
Number of Entries	4

#### 9.1.65.2 Interfaces for Class `basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

An LSB conforming implementation shall provide the generic methods for Class `std::basic_ostringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t>>` specified in Table 9-171, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-171 libstdcxx - Class basic\_ostringstream<wchar\_t, char\_traits<wchar\_t>, allocator<wchar\_t>> Function Interfaces**

<code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::str() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::rdbuf() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ostringstream&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;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::basic_ostringstream(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::basic_ostringstream(_Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::basic_ostringstream(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt; const&amp;, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::basic_ostringstream(_Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::~~basic_ostringstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::~~basic_ostringstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::~~basic_ostringstream()</code> (GLIBCXX_3.4) [ISOCXX]

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 9-172, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-172 libstdcxx - Class basic\_ostringstream<wchar\_t, char\_traits<wchar\_t>, allocator<wchar\_t>> Data Interfaces**

<code>typeid</code> for <code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>typeid</code> name for <code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
VTT for <code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>vtable</code> for <code>basic_ostringstream&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]



## 9.1.66 Class `basic_stringbuf<char, char_traits<char>, allocator<char> >`

### 9.1.66.1 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 9-173

**Table 9-173 Primary vtable for `basic_stringbuf<char, char_traits<char>, allocator<char> >`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;</code>
vfunc[0]:	<code>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::~~basic_stringbuf()</code>
vfunc[1]:	<code>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::~~basic_stringbuf()</code>
vfunc[2]:	<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::imbue(locale const&amp;)</code>
vfunc[3]:	See The Architecture Specific Specification
vfunc[4]:	See The Architecture Specific Specification
vfunc[5]:	<code>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::seekpos(fpos&lt;_mbstate_t&gt;, _Ios_Openmode)</code>
vfunc[6]:	<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::sync()</code>
vfunc[7]:	<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::showmanyc()</code>
vfunc[8]:	See The Architecture Specific Specification
vfunc[9]:	<code>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::underflow()</code>
vfunc[10]:	<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::uflow()</code>
vfunc[11]:	<code>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;::pbackfail(int)</code>

vfunc[12]:	See The Architecture Specific Specification
vfunc[13]:	basic_stringbuf<char, char_traits<char>, allocator<char>>::overflow(int)

The Run Time Type Information for the std::basic\_stringbuf<char, std::char\_traits<char>, std::allocator<char>> class is described by Table 9-174

**Table 9-174 typeid for basic\_stringbuf<char, char\_traits<char>, allocator<char>>**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for basic_stringbuf<char, char_traits<char>, allocator<char>>

### 9.1.66.2 Interfaces for Class basic\_stringbuf<char, char\_traits<char>, allocator<char>>

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 9-175, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-175 libstdc++ - Class basic\_stringbuf<char, char\_traits<char>, allocator<char>> Function Interfaces**

basic_stringbuf<char, char_traits<char>, allocator<char>>::str() const(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char>> >::_M_update_egptr()(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char>> >::_M_stringbuf_init(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char>> >::str(basic_string<char, char_traits<char>, allocator<char>> const&)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char>> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char>> >::overflow(int)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char>> >::pbackfail(int)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char>> >::underflow()(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char>> >::basic_stringbuf(basic_string<char, char_traits<char>, allocator<char>> const&, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]

<code>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::basic_stringbuf(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::basic_stringbuf(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &amp; const&amp;, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::basic_stringbuf(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::~~basic_stringbuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;::~~basic_stringbuf()(GLIBCXX_3.4) [ISOCXX]</code>

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 9-176, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-176 libstdc++ - Class `basic_stringbuf<char, char_traits<char>, allocator<char>>` Data Interfaces**

<code>typeid for basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for basic_stringbuf&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.67 Class `basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`

#### 9.1.67.1 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 9-177

**Table 9-177 Primary vtable for `basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t>>`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid for basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;</code>
<code>vfunc[0]:</code>	<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt;&gt;::~~basic_stringbuf()</code>

vfunc[1]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringbuf()
vfunc[2]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::imbue(locale const&)
vfunc[3]:	See The Architecture Specific Specification
vfunc[4]:	See The Architecture Specific Specification
vfunc[5]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::sync()
vfunc[7]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::showmanyc()
vfunc[8]:	See The Architecture Specific Specification
vfunc[9]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::underflow()
vfunc[10]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::uflow()
vfunc[11]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::pbackfail(unsigned int)
vfunc[12]:	See The Architecture Specific Specification
vfunc[13]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::overflow(unsigned int)

The Run Time Type Information for the std::basic\_stringbuf<wchar\_t, std::char\_traits<wchar\_t>, std::allocator<wchar\_t> > class is described by Table 9-178

**Table 9-178 typeid for basic\_stringbuf<wchar\_t, char\_traits<wchar\_t>, allocator<wchar\_t> >**

Base Vtable	vtable for
-------------	------------

	<code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>basic_stringbuf&lt;wchar_t,</code> <code>char_traits&lt;wchar_t&gt;,</code> <code>allocator&lt;wchar_t&gt; &gt;</code>

### 9.1.67.2 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, char_traits<wchar_t>, allocator<wchar_t> >` specified in Table 9-179, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-179 libstdc++ - Class `basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >` Function Interfaces**

<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::str()</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::M_update_egptr()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::M_stringbuf_init(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&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) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::seekpos(fpos&lt;__mbstate_t&gt;, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::overflow(unsigned int)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::pbackfail(unsigned int)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::underflow()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_stringbuf(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_stringbuf(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_stringbuf(basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::basic_stringbuf(_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;::~~basic_stringbuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;</code>

>::~basic_stringbuf()(GLIBCXX_3.4) [ISOCXX]
---

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 9-180, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-180 libstdcxx - Class `basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >` Data Interfaces**

typename for <code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
--

typename name for <code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
---

vtable for <code>basic_stringbuf&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
--

### 9.1.68 Class `basic_istream<char, char_traits<char> >`

#### 9.1.68.1 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 ISO/IEC 23360.

The VTT for the `std::basic_istream<char, std::char_traits<char> >` class is described by Table 9-181

**Table 9-181 VTT for `basic_istream<char, char_traits<char> >`**

VTT Name	<code>_ZTTSD</code>
Number of Entries	7

#### 9.1.68.2 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 9-182, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-182 libstdcxx - Class `basic_istream<char, char_traits<char> >` Function Interfaces**

<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::basic_istream(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;*)(GLIBCXX_3.4)</code> [ISOCXX]
---

<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::basic_istream()(GLIBCXX_3.4)</code> [ISOCXX]
---

<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::basic_istream(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;*)(GLIBCXX_3.4)</code> [ISOCXX]
---

<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::basic_istream()(GLIBCXX_3.4)</code> [ISOCXX]
---

<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::~basic_istream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::~basic_istream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::~basic_istream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, signed char*)</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_istream<char, std::char_traits<char> >` specified in Table 9-183, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-183 libstdcxx - Class `basic_istream<char, char_traits<char> >` Data Interfaces**

<code>typename for basic_istream&lt;char, char_traits&lt;char&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>typename name for basic_istream&lt;char, char_traits&lt;char&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
VTT for <code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>vtable for basic_istream&lt;char, char_traits&lt;char&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.69 Class `basic_istream<wchar_t, char_traits<wchar_t> >`

#### 9.1.69.1 Class data for `basic_istream<wchar_t, char_traits<wchar_t> >`

The virtual table for the `std::basic_istream<wchar_t, std::char_traits<wchar_t> >` class is described in the relevant architecture specific part of ISO/IEC 23360.

The VTT for the `std::basic_istream<wchar_t, std::char_traits<wchar_t> >` class is described by Table 9-184

**Table 9-184 VTT for `basic_istream<wchar_t, char_traits<wchar_t> >`**

VTT Name	<code>_ZTTSt14basic_istreamIwSt11char_traitsIwEE</code>
Number of Entries	7

#### 9.1.69.2 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 9-185, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-185 libstdcxx - Class basic\_iostream<wchar\_t, char\_traits<wchar\_t> > Function Interfaces**

basic_iostream<wchar_t, char_traits<wchar_t> >::basic_iostream(basic_streambuf<wchar_t, char_traits<wchar_t> >*)(GLIBCXX_3.4) [ISOCXX]
basic_iostream<wchar_t, char_traits<wchar_t> >::basic_iostream()(GLIBCXX_3.4) [ISOCXX]
basic_iostream<wchar_t, char_traits<wchar_t> >::basic_iostream(basic_streambuf<wchar_t, char_traits<wchar_t> >*)(GLIBCXX_3.4) [ISOCXX]
basic_iostream<wchar_t, char_traits<wchar_t> >::basic_iostream()(GLIBCXX_3.4) [ISOCXX]
basic_iostream<wchar_t, char_traits<wchar_t> >::~~basic_iostream()(GLIBCXX_3.4) [ISOCXX]
basic_iostream<wchar_t, char_traits<wchar_t> >::~~basic_iostream()(GLIBCXX_3.4) [ISOCXX]
basic_iostream<wchar_t, char_traits<wchar_t> >::~~basic_iostream()(GLIBCXX_3.4) [ISOCXX]

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 9-186, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-186 libstdcxx - Class basic\_iostream<wchar\_t, char\_traits<wchar\_t> > Data Interfaces**

typeid for basic_iostream<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]
typeid name for basic_iostream<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]
VTT for basic_iostream<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]
vtable for basic_iostream<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]

### 9.1.70 Class basic\_istream<char, char\_traits<char> >

#### 9.1.70.1 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 ISO/IEC 23360.

The VTT for the `std::basic_istream<char, std::char_traits<char> >` class is described by Table 9-187

**Table 9-187 VTT for basic\_istream<char, char\_traits<char> >**

VTT Name	_ZTTSi
Number of Entries	2



### 9.1.70.2 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 9-188, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-188 libstdcxx - Class `basic_istream<char, char_traits<char>>` > Function Interfaces**

<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::gcount() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::sentry::operator bool() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::get(basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;&amp;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::get(basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;&amp;, char)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::get(char&amp;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::get()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::peek()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::sync()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::seekg(fpos&lt;__mbstate_t&gt;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::tellg()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::unget()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::sentry::sentry(basic_istream&lt;char, char_traits&lt;char&gt;&gt;&amp;, bool)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::sentry::sentry(basic_istream&lt;char, char_traits&lt;char&gt;&gt;&amp;, bool)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::putback(char)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::basic_istream(basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::basic_istream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::basic_istream(basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::basic_istream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_istream&lt;char, char_traits&lt;char&gt;&gt;&gt;::~basic_istream()</code> (GLIBCXX_3.4) [ISOCXX]

<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::~basic_istream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::~basic_istream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>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;))(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(ios_base&amp; (*)(ios_base&amp;))(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(basic_ios&lt;char, char_traits&lt;char&gt; &gt;&amp; (*)(basic_ios&lt;char, char_traits&lt;char&gt; &gt;&amp;))(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(void*&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(bool&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(double&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(long double&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(float&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(int&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(unsigned int&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(long&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(unsigned long&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(short&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(unsigned short&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(long long&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;::operator&gt;&gt;(unsigned long long&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>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;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; getline&lt;char, char_traits&lt;char&gt;,</code>

<p>allocator&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; getline&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&amp;, char)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, unsigned char*)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, signed char&amp;)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, unsigned char&amp;)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, char*)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, char&amp;)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Setiosflags)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Setprecision)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Resetiosflags)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Setw)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Setbase)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Setfill&lt;char&gt;)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</p>
<p>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;double, char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;,</p>

<code>complex&lt;double&gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;long double, char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, complex&lt;long double&gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&gt;&gt;&lt;float, char, char_traits&lt;char&gt; &gt;(basic_istream&lt;char, char_traits&lt;char&gt; &gt;&amp;, complex&lt;float&gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_istream<char, std::char_traits<char> >` specified in Table 9-189, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-189 libstdcxx - Class `basic_istream<char, char_traits<char> >` Data Interfaces**

<code>typename for basic_istream&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typename name for basic_istream&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>VTT for basic_istream&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for basic_istream&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.71 Class `basic_istream<wchar_t, char_traits<wchar_t> >`

#### 9.1.71.1 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 ISO/IEC 23360.

The VTT for the `std::basic_istream<wchar_t, std::char_traits<wchar_t> >` class is described by Table 9-190

**Table 9-190 VTT for `basic_istream<wchar_t, char_traits<wchar_t> >`**

VTT Name	<code>_ZTTSt13basic_istreamIwSt11char_traitsIwEE</code>
Number of Entries	2

#### 9.1.71.2 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 9-191, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-191 libstdcxx - Class `basic_istream<wchar_t, char_traits<wchar_t> >` Function Interfaces**

<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::gcount() const(GLIBCXX_3.4) [ISOCXX]</code>
--

<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::sentry::operator bool() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::get(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::get(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, wchar_t)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::get(wchar_t&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::get()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::peek()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::sync()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::seekg(fpos&lt;__mbstate_t&gt;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::tellg()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::unget()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::sentry::sentry(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, bool)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::sentry::sentry(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, bool)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::putback(wchar_t)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_istream(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_istream(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~~basic_istream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~~basic_istream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~~basic_istream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;</code>

<code>(*)(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;))(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(ios_base&amp; (*) (ios_base&amp;))(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; (*) (basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;))(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;*) (GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(void*&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(bool&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(double&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(long double&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(float&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(int&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(unsigned int&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(long&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(unsigned long&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(short&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(unsigned short&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(long long&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&gt;&gt;(unsigned long long&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; ws&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>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;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>

<code>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;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;&amp;, wchar_t)(GLIBCXX_3.4) [ISOCXX]</code>
<code>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; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, complex&lt;double&gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>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; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, complex&lt;long double&gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>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; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, complex&lt;float&gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, wchar_t*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, wchar_t&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, _Setiosflags)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, _Setprecision)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, _Resetiosflags)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, _Setw)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, _Setbase)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, _Setfill&lt;wchar_t&gt;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&gt;&gt;&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;(basic_istream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>

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 9-192, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-192 libstdcxx - Class basic\_istream<wchar\_t, char\_traits<wchar\_t> > Data Interfaces**

typeid for basic_istream<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]
typeid name for basic_istream<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]
VTT for basic_istream<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]
vtbl for basic_istream<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]

### 9.1.72 Class istreambuf\_iterator<wchar\_t, char\_traits<wchar\_t> >

#### 9.1.72.1 Interfaces for Class istreambuf\_iterator<wchar\_t, char\_traits<wchar\_t> >

No external methods are defined for libstdcxx - Class std::istreambuf\_iterator<wchar\_t, std::char\_traits<wchar\_t> > in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

### 9.1.73 Class istreambuf\_iterator<char, char\_traits<char> >

#### 9.1.73.1 Interfaces for Class istreambuf\_iterator<char, char\_traits<char> >

No external methods are defined for libstdcxx - Class std::istreambuf\_iterator<char, std::char\_traits<char> > in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

### 9.1.74 Class basic\_ostream<char, char\_traits<char> >

#### 9.1.74.1 Class data for basic\_ostream<char, char\_traits<char> >

The virtual table for the std::basic\_ostream<char, std::char\_traits<char> > class is described in the relevant architecture specific part of ISO/IEC 23360.

The VTT for the std::basic\_ostream<char, std::char\_traits<char> > class is described by Table 9-193

**Table 9-193 VTT for basic\_ostream<char, char\_traits<char> >**

VTT Name	_ZTTSo
Number of Entries	2

#### 9.1.74.2 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 9-194, with the full mandatory functionality as described in the referenced underlying specification.



**Table 9-194 libstdcxx - Class basic\_ostream<char, char\_traits<char> > Function Interfaces**

basic_ostream<char, char_traits<char> >::sentry::operator bool() const(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::put(char)(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::flush()(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::seekp(fpos<__mbstate_t>)(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::tellp()(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::sentry::sentry(basic_ostream<char, char_traits<char> >&)(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::sentry::sentry(basic_ostream<char, char_traits<char> >&)(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::sentry::~sentry()(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::sentry::~sentry()(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::basic_ostream(basic_streambuf<char, char_traits<char> >*)(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::basic_ostream()(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::basic_ostream(basic_streambuf<char, char_traits<char> >*)(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::basic_ostream()(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::~basic_ostream()(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::~basic_ostream()(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::~basic_ostream()(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::operator<<(basic_ostream<char, char_traits<char> >& (*)(basic_ostream<char, char_traits<char> >&))(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::operator<<(ios_base& (*)(ios_base&))(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::operator<<(basic_ios<char, char_traits<char> >& (*)(basic_ios<char, char_traits<char> >&))(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::operator<<(void const*)(GLIBCXX_3.4) [ISOCXX]

<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(bool)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(double)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(long double)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(float)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(int)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(unsigned int)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(short)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(unsigned short)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(long long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;::operator&lt;&lt;(unsigned long long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; endl&lt;char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; ends&lt;char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; flush&lt;char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, signed char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, unsigned char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, signed char)(GLIBCXX_3.4)</code>

[ISOCXX]
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, char)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, unsigned char)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Setiosflags)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Setprecision)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Resetiosflags)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Setw)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Setbase)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, _Setfill&lt;char&gt;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;double, char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, complex&lt;double&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;long double, char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, complex&lt;long double&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp; operator&lt;&lt; &lt;float, char, char_traits&lt;char&gt; &gt;(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;&amp;, complex&lt;float&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_ostream<char, std::char_traits<char> >` specified in Table 9-195, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-195 libstdcxx - Class `basic_ostream<char, char_traits<char> >` Data Interfaces**

<code>typeid for basic_ostream&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4)</code>
--

[CXXABI]
typeid name for basic_ostream<char, char_traits<char> >(GLIBCXX_3.4) [CXXABI]
VTT for basic_ostream<char, char_traits<char> >(GLIBCXX_3.4) [CXXABI]
vtable for basic_ostream<char, char_traits<char> >(GLIBCXX_3.4) [CXXABI]

### 9.1.75 Class basic\_ostream<wchar\_t, char\_traits<wchar\_t> >

#### 9.1.75.1 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 ISO/IEC 23360.

The VTT for the std::basic\_ostream<wchar\_t, std::char\_traits<wchar\_t> > class is described by Table 9-196

**Table 9-196 VTT for basic\_ostream<wchar\_t, char\_traits<wchar\_t> >**

VTT Name	_ZTTSt13basic_ostreamIwSt11char_traitsIwEE
Number of Entries	2

#### 9.1.75.2 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 9-197, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-197 libstdcxx - Class basic\_ostream<wchar\_t, char\_traits<wchar\_t> > Function Interfaces**

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]

<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::sentry::~sentry()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::sentry::~sentry()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::basic_ostream(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::basic_ostream(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::~~basic_ostream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::~~basic_ostream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::~~basic_ostream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::operator&lt;&lt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt; &amp;&amp; (*) (basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt; &amp;&amp;))(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::operator&lt;&lt;(ios_base&amp; (*) (ios_base&amp;))(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::operator&lt;&lt;(basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt; &amp;&amp; (*) (basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt; &amp;&amp;))(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::operator&lt;&lt;(void const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::operator&lt;&lt;(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::operator&lt;&lt;(bool)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::operator&lt;&lt;(double)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::operator&lt;&lt;(long double)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::operator&lt;&lt;(float)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::operator&lt;&lt;(int)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::operator&lt;&lt;(unsigned int)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</code>

<code>&gt;::operator&lt;&lt;(long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&lt;&lt;(unsigned long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&lt;&lt;(short)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&lt;&lt;(unsigned short)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&lt;&lt;(long long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator&lt;&lt;(unsigned long long)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; endl&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; ends&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; flush&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;double, wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, complex&lt;double&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;long double, wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, complex&lt;long double&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;float, wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, complex&lt;float&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, wchar_t const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, wchar_t)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, _Setiosflags)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;,</code>

<code>_Setprecision)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, _Resetiosflags)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, _Setw)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, _Setbase)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, _Setfill&lt;wchar_t&gt;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, char)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp; operator&lt;&lt; &lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&amp;, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>

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 9-198, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-198 libstdcxx - Class `basic_ostream<wchar_t, char_traits<wchar_t> >` Data Interfaces**

<code>typeid for basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>VTT for basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.76 Class `basic_fstream<char, char_traits<char> >`

#### 9.1.76.1 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 ISO/IEC 23360.

The VTT for the `std::basic_fstream<char, std::char_traits<char> >` class is described by Table 9-199

**Table 9-199 VTT for `basic_fstream<char, char_traits<char> >`**

VTT Name	<code>_ZTTSt13basic_fstreamIcSt11char_tra</code>
----------	--

	itsIcEE
Number of Entries	10

### 9.1.76.2 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 9-200, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-200 libstdcxx - Class `basic_fstream<char, char_traits<char>>` > Function Interfaces**

<code>basic_fstream&lt;char, char_traits&lt;char&gt;&gt;::rdbuf() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_fstream&lt;char, char_traits&lt;char&gt;&gt;::open(char const*, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_fstream&lt;char, char_traits&lt;char&gt;&gt;::close()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_fstream&lt;char, char_traits&lt;char&gt;&gt;::is_open()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_fstream&lt;char, char_traits&lt;char&gt;&gt;::basic_fstream(char const*, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_fstream&lt;char, char_traits&lt;char&gt;&gt;::basic_fstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_fstream&lt;char, char_traits&lt;char&gt;&gt;::basic_fstream(char const*, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_fstream&lt;char, char_traits&lt;char&gt;&gt;::basic_fstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_fstream&lt;char, char_traits&lt;char&gt;&gt;::~~basic_fstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_fstream&lt;char, char_traits&lt;char&gt;&gt;::~~basic_fstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_fstream&lt;char, char_traits&lt;char&gt;&gt;::~~basic_fstream()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_fstream<char, std::char_traits<char>>` specified in Table 9-201, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-201 libstdcxx - Class `basic_fstream<char, char_traits<char>>` > Data Interfaces**

<code>typeid for basic_fstream&lt;char, char_traits&lt;char&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>typeid name for basic_fstream&lt;char, char_traits&lt;char&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>VTT for basic_fstream&lt;char, char_traits&lt;char&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]



vtable for basic_fstream<char, char_traits<char> >(GLIBCXX_3.4) [CXXABI]
--

### 9.1.77 Class basic\_fstream<wchar\_t, char\_traits<wchar\_t> >

#### 9.1.77.1 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 ISO/IEC 23360.

The VTT for the std::basic\_fstream<wchar\_t, std::char\_traits<wchar\_t> > class is described by Table 9-202

**Table 9-202 VTT for basic\_fstream<wchar\_t, char\_traits<wchar\_t> >**

VTT Name	_ZTTSt13basic_fstreamlwSt11char_traitslwEE
Number of Entries	10

#### 9.1.77.2 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 9-203, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-203 libstdc++ - Class basic\_fstream<wchar\_t, char\_traits<wchar\_t> > Function Interfaces**

basic_fstream<wchar_t, char_traits<wchar_t> >::rdbuf() const(GLIBCXX_3.4) [ISOCXX]
basic_fstream<wchar_t, char_traits<wchar_t> >::open(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_fstream<wchar_t, char_traits<wchar_t> >::close()(GLIBCXX_3.4) [ISOCXX]
basic_fstream<wchar_t, char_traits<wchar_t> >::is_open()(GLIBCXX_3.4) [ISOCXX]
basic_fstream<wchar_t, char_traits<wchar_t> >::basic_fstream(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_fstream<wchar_t, char_traits<wchar_t> >::basic_fstream()(GLIBCXX_3.4) [ISOCXX]
basic_fstream<wchar_t, char_traits<wchar_t> >::basic_fstream(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_fstream<wchar_t, char_traits<wchar_t> >::basic_fstream()(GLIBCXX_3.4) [ISOCXX]
basic_fstream<wchar_t, char_traits<wchar_t> >::~~basic_fstream()(GLIBCXX_3.4) [ISOCXX]
basic_fstream<wchar_t, char_traits<wchar_t> >::~~basic_fstream()(GLIBCXX_3.4) [ISOCXX]

<pre>basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~basic_fstream()(GLIBCXX_3.4) [ISOCXX]</pre>
--

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 9-204, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-204 libstdcxx - Class `basic_fstream<wchar_t, char_traits<wchar_t> >` Data Interfaces**

<pre>typename for basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</pre>
<pre>typename name for basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</pre>
<pre>VTT for basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</pre>
<pre>vtable for basic_fstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</pre>

### 9.1.78 Class `basic_ifstream<char, char_traits<char> >`

#### 9.1.78.1 Class data for `basic_ifstream<char, char_traits<char> >`

The virtual table for the `std::basic_ifstream<char, std::char_traits<char> >` class is described in the relevant architecture specific part of ISO/IEC 23360.

The VTT for the `std::basic_ifstream<char, std::char_traits<char> >` class is described by Table 9-205

**Table 9-205 VTT for `basic_ifstream<char, char_traits<char> >`**

VTT Name	<code>_ZTTSt14basic_ifstreamIcSt11char_traitsIcEE</code>
Number of Entries	4

#### 9.1.78.2 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 9-206, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-206 libstdcxx - Class `basic_ifstream<char, char_traits<char> >` Function Interfaces**

<pre>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::rdbuf() const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::open(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::close()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::is_open()(GLIBCXX_3.4) [ISOCXX]</pre>

<code>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ifstream(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ifstream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ifstream(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::basic_ifstream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::~~basic_ifstream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::~~basic_ifstream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;::~~basic_ifstream()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_ifstream<char, std::char_traits<char> >` specified in Table 9-207, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-207 libstdcxx - Class `basic_ifstream<char, char_traits<char> >` Data Interfaces**

<code>typeid for basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>VTT for basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for basic_ifstream&lt;char, char_traits&lt;char&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.79 Class `basic_ifstream<wchar_t, char_traits<wchar_t> >`

#### 9.1.79.1 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 ISO/IEC 23360.

The VTT for the `std::basic_ifstream<wchar_t, std::char_traits<wchar_t> >` class is described by Table 9-208

**Table 9-208 VTT for `basic_ifstream<wchar_t, char_traits<wchar_t> >`**

VTT Name	<code>_ZTTSt14basic_ifstreamIwSt11char_t raitsIwEE</code>
Number of Entries	4

### 9.1.79.2 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 9-209, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-209 libstdcxx - Class `basic_ifstream<wchar_t, char_traits<wchar_t>>` Function Interfaces**

<code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::rdbuf() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::open(char const*, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::close()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::is_open()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::basic_ifstream(char const*, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::basic_ifstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::basic_ifstream(char const*, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::basic_ifstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::~~basic_ifstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::~~basic_ifstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::~~basic_ifstream()</code> (GLIBCXX_3.4) [ISOCXX]

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 9-210, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-210 libstdcxx - Class `basic_ifstream<wchar_t, char_traits<wchar_t>>` Data Interfaces**

<code>typeid for basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>typeid for basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>typeid name for basic_ifstream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]

typeinfo name for basic_streambuf<wchar_t, char_traits<wchar_t>>(GLIBCXX_3.4) [CXXABI]
VTT for basic_ifstream<wchar_t, char_traits<wchar_t>>(GLIBCXX_3.4) [CXXABI]
vtable for basic_ifstream<wchar_t, char_traits<wchar_t>>(GLIBCXX_3.4) [CXXABI]
vtable for basic_streambuf<wchar_t, char_traits<wchar_t>>(GLIBCXX_3.4) [CXXABI]

### 9.1.80 Class basic\_ofstream<char, char\_traits<char>>

#### 9.1.80.1 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 ISO/IEC 23360.

The VTT for the std::basic\_ofstream<char, std::char\_traits<char>> class is described by Table 9-211

**Table 9-211 VTT for basic\_ofstream<char, char\_traits<char>>**

VTT Name	_ZTTSt14basic_ofstreamIcSt11char_traitsIcEE
Number of Entries	4

#### 9.1.80.2 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 9-212, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-212 libstdcxx - Class basic\_ofstream<char, char\_traits<char>> Function Interfaces**

basic_ofstream<char, char_traits<char>>::rdbuf() const(GLIBCXX_3.4) [ISOCXX]
basic_ofstream<char, char_traits<char>>::open(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_ofstream<char, char_traits<char>>::close()(GLIBCXX_3.4) [ISOCXX]
basic_ofstream<char, char_traits<char>>::is_open()(GLIBCXX_3.4) [ISOCXX]
basic_ofstream<char, char_traits<char>>::basic_ofstream(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_ofstream<char, char_traits<char>>::basic_ofstream()(GLIBCXX_3.4) [ISOCXX]
basic_ofstream<char, char_traits<char>>::basic_ofstream(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_ofstream<char, char_traits<char>>::basic_ofstream()(GLIBCXX_3.4) [ISOCXX]

<code>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::~basic_ofstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::~basic_ofstream()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;::~basic_ofstream()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_ofstream<char, std::char_traits<char> >` specified in Table 9-213, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-213 libstdcxx - Class `basic_ofstream<char, char_traits<char> >` Data Interfaces**

<code>typeid</code> for <code>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>typeid</code> name for <code>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
VTT for <code>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
<code>vtable</code> for <code>basic_ofstream&lt;char, char_traits&lt;char&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.81 Class `basic_ofstream<wchar_t, char_traits<wchar_t> >`

#### 9.1.81.1 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 ISO/IEC 23360.

The VTT for the `std::basic_ofstream<wchar_t, std::char_traits<wchar_t> >` class is described by Table 9-214

**Table 9-214 VTT for `basic_ofstream<wchar_t, char_traits<wchar_t> >`**

VTT Name	<code>_ZTTSt14basic_ofstreamIwSt11char_traitsIwEE</code>
Number of Entries	4

#### 9.1.81.2 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 9-215, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-215 libstdcxx - Class `basic_ofstream<wchar_t, char_traits<wchar_t> >` Function Interfaces**

<code>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::rdbuf()</code> <code>const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::open(char const*,</code>

<code>_Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::close()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::is_open()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_ofstream(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_ofstream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_ofstream(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_ofstream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~~basic_ofstream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~~basic_ofstream()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~~basic_ofstream()(GLIBCXX_3.4) [ISOCXX]</code>

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 9-216, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-216 libstdcxx - Class `basic_ofstream<wchar_t, char_traits<wchar_t> >` Data Interfaces**

<code>typeid for basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>VTT for basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for basic_ofstream&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>

## 9.1.82 Class `basic_streambuf<char, char_traits<char> >`

### 9.1.82.1 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 9-217

**Table 9-217 Primary vtable for `basic_streambuf<char, char_traits<char> >`**

Base Offset	0
Virtual Base Offset	0

RTTI	typeinfo for basic_streambuf<char, char_traits<char> >
vfunc[0]:	basic_streambuf<char, char_traits<char> >::~~basic_streambuf()
vfunc[1]:	basic_streambuf<char, char_traits<char> >::~~basic_streambuf()
vfunc[2]:	basic_streambuf<char, char_traits<char> >::imbue(locale const&)
vfunc[3]:	See The Architecture Specific Specification
vfunc[4]:	See The Architecture Specific Specification
vfunc[5]:	basic_streambuf<char, char_traits<char> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_streambuf<char, char_traits<char> >::sync()
vfunc[7]:	basic_streambuf<char, char_traits<char> >::showmanyc()
vfunc[8]:	See The Architecture Specific Specification
vfunc[9]:	basic_streambuf<char, char_traits<char> >::underflow()
vfunc[10]:	basic_streambuf<char, char_traits<char> >::uflow()
vfunc[11]:	basic_streambuf<char, char_traits<char> >::pbackfail(int)
vfunc[12]:	See The Architecture Specific Specification
vfunc[13]:	basic_streambuf<char, char_traits<char> >::overflow(int)

The Run Time Type Information for the std::basic\_streambuf<char, std::char\_traits<char> > class is described by Table 9-218

**Table 9-218 typeinfo for basic\_streambuf<char, char\_traits<char> >**

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for basic_streambuf<char,



	char_traits<char> >
--	---------------------

### 9.1.82.2 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 9-219, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-219 libstdcxx - Class basic\_streambuf<char, char\_traits<char> > Function Interfaces**

basic_streambuf<char, char_traits<char> >::gptr() const(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::pptr() const(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::eback() const(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::egptr() const(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::epptr() const(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::pbase() const(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::getloc() const(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::pubseekpos(fpos<__mbstate_t>, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::setg(char*, char*, char*)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::setp(char*, char*)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::sync()(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::gbump(int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::imbue(locale const&)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::pbump(int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::sgetc()(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::sputc(char)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::uflow()(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::sbumpc()(GLIBCXX_3.4)

[ISOCXX]
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::snextc()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::pubsync()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::seekpos(fpos&lt;__mbstate_t&gt;, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::sungetc()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::in_avail()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::overflow(int)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::pubimbue(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::pbackfail(int)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::showmanyc()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::sputbackc(char)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::underflow()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::basic_streambuf(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::basic_streambuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::basic_streambuf(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::basic_streambuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::~~basic_streambuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::~~basic_streambuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::~~basic_streambuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;::operator=(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_streambuf<char, std::char_traits<char> >` specified in Table 9-

220, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-220 libstdcxx - Class `basic_streambuf<char, char_traits<char>>` > Data Interfaces**

typeinfo for <code>basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
typeinfo name for <code>basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>basic_streambuf&lt;char, char_traits&lt;char&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.83 Class `basic_streambuf<wchar_t, char_traits<wchar_t>>`

#### 9.1.83.1 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 9-221

**Table 9-221 Primary vtable for `basic_streambuf<wchar_t, char_traits<wchar_t>>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</code>
vfunc[0]:	<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt; &gt;::~~basic_streambuf()</code>
vfunc[1]:	<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt; &gt;::~~basic_streambuf()</code>
vfunc[2]:	<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::imbue(locale const&amp;)</code>
vfunc[3]:	See The Architecture Specific Specification
vfunc[4]:	See The Architecture Specific Specification
vfunc[5]:	<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt; &gt;::seekpos(fpos&lt;__mbstate_t&gt;, _ios_Openmode)</code>
vfunc[6]:	<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::sync()</code>

vfunc[7]:	basic_streambuf<wchar_t, char_traits<wchar_t>>::showmanyc()
vfunc[8]:	See The Architecture Specific Specification
vfunc[9]:	basic_streambuf<wchar_t, char_traits<wchar_t>>::underflow()
vfunc[10]:	basic_streambuf<wchar_t, char_traits<wchar_t>>::uflow()
vfunc[11]:	basic_streambuf<wchar_t, char_traits<wchar_t>>::pbackfail(unsigned int)
vfunc[12]:	See The Architecture Specific Specification
vfunc[13]:	basic_streambuf<wchar_t, char_traits<wchar_t>>::overflow(unsigned int)

The Run Time Type Information for the `std::basic_streambuf<wchar_t, std::char_traits<wchar_t>>` class is described by Table 9-222

**Table 9-222 typeid for `basic_streambuf<wchar_t, char_traits<wchar_t>>`**

Base Vtable	vtable for <code>__cxxabiv1::__class_type_info</code>
Name	typeid name for <code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;</code>

### 9.1.83.2 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 9-223, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-223 `libstdcxx` - Class `basic_streambuf<wchar_t, char_traits<wchar_t>>` > Function Interfaces**

<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::gpptr()</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::ppptr()</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::eback()</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::egpptr()</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::eppptr()</code>

const(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::pbase() const(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::getloc() const(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::pubseekpos(fpos<__mbstate_t>, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::setg(wchar_t*, wchar_t*, wchar_t*)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::setp(wchar_t*, wchar_t*)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::sync()(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::gbump(int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::imbue(locale const&)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::pbump(int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::sgetc()(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::sputc(wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::uflow()(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::sbumpc()(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::sngetc()(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::pubsync()(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::sungetc()(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::in_avail()(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::overflow(unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::pubimbue(locale const&)(GLIBCXX_3.4) [ISOCXX]

<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::pbackfail(unsigned int)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::showmanyc()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::sputbackc(wchar_t)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::underflow()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_streambuf(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_streambuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_streambuf(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::basic_streambuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~~basic_streambuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~~basic_streambuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::~~basic_streambuf()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::operator=(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>

### 9.1.84 Class `basic_filebuf<char, char_traits<char> >`

#### 9.1.84.1 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 9-224

**Table 9-224 Primary vtable for `basic_filebuf<char, char_traits<char> >`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeinfo for basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;</code>
<code>vfunc[0]:</code>	<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::~~basic_filebuf()</code>
<code>vfunc[1]:</code>	<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::~~basic_filebuf()</code>
<code>vfunc[2]:</code>	<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;</code>

	>::imbue(locale const&)
vfunc[3]:	See The Architecture Specific Specification
vfunc[4]:	See The Architecture Specific Specification
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 The Architecture Specific Specification
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 The Architecture Specific Specification
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 9-225

**Table 9-225 typeid for `basic_filebuf<char, char_traits<char>>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>basic_filebuf&lt;char, char_traits&lt;char&gt;&gt;</code>

### 9.1.84.2 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 9-226, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-226 libstdc++ - Class `basic_filebuf<char, char_traits<char>>` Function Interfaces**

<code>basic_filebuf&lt;char, char_traits&lt;char&gt;&gt;::is_open() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt;&gt;::M_create_pback()</code> (GLIBCXX_3.4)

[ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::_M_destroy_pback()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::_M_terminate_output()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::_M_destroy_internal_buffer()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::_M_allocate_internal_buffer()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::open(char const*, _Ios_Openmode)(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::sync()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::close()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::imbue(locale const&amp;)(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::seekpos(fpos&lt;__mbstate_t&gt;, _Ios_Openmode)(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::overflow(int)(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::pbackfail(int)(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::showmanyc()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::underflow()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::basic_filebuf()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::basic_filebuf()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::~~basic_filebuf()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::~~basic_filebuf()(GLIBCXX_3.4)</code> [ISOCXX]
<code>basic_filebuf&lt;char, char_traits&lt;char&gt; &gt;::~~basic_filebuf()(GLIBCXX_3.4)</code> [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_filebuf<char, std::char_traits<char> >` specified in Table 9-227, with the full mandatory functionality as described in the referenced underlying specification.



**Table 9-227 libstdcxx - Class basic\_filebuf<char, char\_traits<char> > Data Interfaces**

typeid for basic_filebuf<char, char_traits<char> >(GLIBCXX_3.4) [CXXABI]
typeid name for basic_filebuf<char, char_traits<char> >(GLIBCXX_3.4) [CXXABI]
vtable for basic_filebuf<char, char_traits<char> >(GLIBCXX_3.4) [CXXABI]

**9.1.85 Class basic\_filebuf<wchar\_t, char\_traits<wchar\_t> >****9.1.85.1 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 9-228

**Table 9-228 Primary vtable for basic\_filebuf<wchar\_t, char\_traits<wchar\_t> >**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for basic_filebuf<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::~~basic_filebuf()
vfunc[1]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::~~basic_filebuf()
vfunc[2]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::imbue(locale const&)
vfunc[3]:	See The Architecture Specific Specification
vfunc[4]:	See The Architecture Specific Specification
vfunc[5]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::sync()
vfunc[7]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::showmanyc()
vfunc[8]:	See The Architecture Specific Specification
vfunc[9]:	basic_filebuf<wchar_t,

	<code>char_traits&lt;wchar_t&gt; &gt;::underflow()</code>
<code>vfunc[10]:</code>	<code>basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::uflow()</code>
<code>vfunc[11]:</code>	<code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::pbackfail(unsigned int)</code>
<code>vfunc[12]:</code>	See The Architecture Specific Specification
<code>vfunc[13]:</code>	<code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::overflow(unsigned int)</code>

The Run Time Type Information for the `std::basic_filebuf<wchar_t, std::char_traits<wchar_t> >` class is described by Table 9-229

**Table 9-229** typeid for `basic_filebuf<wchar_t, char_traits<wchar_t> >`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;</code>

### 9.1.85.2 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 9-230, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-230** `libstdc++` - Class `basic_filebuf<wchar_t, char_traits<wchar_t> >` Function Interfaces

<code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::is_open()</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::_M_create_pback()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::_M_destroy_pback()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::_M_terminate_output()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::_M_destroy_internal_buffer()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::_M_allocate_internal_buffer()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::open(char const*, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;::sync()(GLIBCXX_3.4)</code>

[ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::close()(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::imbue(locale const&)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::overflow(unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::pbackfail(unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::showmanyc()(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::underflow()(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::basic_filebuf()(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::basic_filebuf()(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::~~basic_filebuf()(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::~~basic_filebuf()(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::~~basic_filebuf()(GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t> >::basic_istream()(GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t> >::basic_istream()(GLIBCXX_3.4) [ISOCXX]
basic_ostream<wchar_t, char_traits<wchar_t> >::basic_ostream()(GLIBCXX_3.4) [ISOCXX]
basic_ostream<wchar_t, char_traits<wchar_t> >::basic_ostream()(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_filebuf<wchar_t, std::char_traits<wchar_t> >` specified in Table 9-231, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-231 libstdcxx - Class `basic_filebuf<wchar_t, char_traits<wchar_t> >` Data Interfaces**

typeid for <code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>basic_filebuf&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;</code>

>(GLIBCXX_3.4) [CXXABI]
vtable for basic_filebuf<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]

## 9.1.86 Class ios\_base

### 9.1.86.1 Class data for ios\_base

The Run Time Type Information for the std::ios\_base class is described by Table 9-232

**Table 9-232 typeinfo for ios\_base**

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for ios_base

### 9.1.86.2 Interfaces for Class ios\_base

An LSB conforming implementation shall provide the generic methods for Class std::ios\_base specified in Table 9-233, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-233 libstdcxx - Class ios\_base Function Interfaces**

ios_base::_M_grow_words(int, bool)(GLIBCXX_3.4) [ISOCXX]
ios_base::sync_with_stdio(bool)(GLIBCXX_3.4) [ISOCXX]
ios_base::register_callback(void (*)(ios_base::event, ios_base&, int), int)(GLIBCXX_3.4) [ISOCXX]
ios_base::Init::Init()(GLIBCXX_3.4) [ISOCXX]
ios_base::Init::Init()(GLIBCXX_3.4) [ISOCXX]
ios_base::Init::~~Init()(GLIBCXX_3.4) [ISOCXX]
ios_base::Init::~~Init()(GLIBCXX_3.4) [ISOCXX]
ios_base::imbue(locale const&)(GLIBCXX_3.4) [ISOCXX]
ios_base::xalloc()(GLIBCXX_3.4) [ISOCXX]
ios_base::_M_init()(GLIBCXX_3.4) [ISOCXX]
ios_base::failure::failure(basic_string<char, char_traits<char>, allocator<char> > const&)(GLIBCXX_3.4) [ISOCXX]
ios_base::failure::failure(basic_string<char, char_traits<char>, allocator<char> > const&)(GLIBCXX_3.4) [ISOCXX]
ios_base::failure::~~failure()(GLIBCXX_3.4) [ISOCXX]
ios_base::failure::~~failure()(GLIBCXX_3.4) [ISOCXX]
ios_base::failure::~~failure()(GLIBCXX_3.4) [ISOCXX]
ios_base::ios_base()(GLIBCXX_3.4) [ISOCXX]
ios_base::ios_base()(GLIBCXX_3.4) [ISOCXX]

<code>ios_base::~ios_base()</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::~ios_base()</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::~ios_base()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::ios_base` specified in Table 9-234, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-234 libstdcxx - Class `ios_base` Data Interfaces**

<code>ios_base::floatfield</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::scientific</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::adjustfield</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::in</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::app</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::ate</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::beg</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::cur</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::dec</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::end</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::hex</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::oct</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::out</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::left</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::fixed</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::right</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::trunc</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::badbit</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::binary</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::eofbit</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::skipws</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::failbit</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::goodbit</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::showpos</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::unitbuf</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::internal</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::showbase</code> (GLIBCXX_3.4) [ISOCXX]
<code>ios_base::basefield</code> (GLIBCXX_3.4) [ISOCXX]

ios_base::boolalpha(GLIBCXX_3.4) [ISOCXX]
ios_base::showpoint(GLIBCXX_3.4) [ISOCXX]
ios_base::uppercase(GLIBCXX_3.4) [ISOCXX]
typeid for ios_base(GLIBCXX_3.4) [CXXABI]
typeid name for ios_base(GLIBCXX_3.4) [CXXABI]
vtable for ios_base(GLIBCXX_3.4) [ISOCXX]

### 9.1.87 Class basic\_ios<char, char\_traits<char> >

#### 9.1.87.1 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 9-235

**Table 9-235 Primary vtable for basic\_ios<char, char\_traits<char> >**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for basic_ios<char, char_traits<char> >
vfunc[0]:	basic_ios<char, char_traits<char> >::~~basic_ios()
vfunc[1]:	basic_ios<char, char_traits<char> >::~~basic_ios()

#### 9.1.87.2 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 9-236, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-236 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> >::rdbuf() const(GLIBCXX_3.4) [ISOCXX]
basic_ios<char, char_traits<char> >::widen(char) const(GLIBCXX_3.4) [ISOCXX]

<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::narrow(char, char) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::rdstate() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::operator void*() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::operator!() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::exceptions(_Ios_Iostate)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::_M_setstate(_Ios_Iostate)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::tie(basic_ostream&lt;char, char_traits&lt;char&gt; &gt;*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::fill(char)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::init(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::clear(_Ios_Iostate)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::imbue(locale const&amp;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::rdbuf(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::copyfmt(basic_ios&lt;char, char_traits&lt;char&gt; &gt; const&amp;)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::setstate(_Ios_Iostate)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::basic_ios(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::basic_ios()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::basic_ios(basic_streambuf&lt;char, char_traits&lt;char&gt; &gt;*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::basic_ios()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::~~basic_ios()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::~~basic_ios()</code> (GLIBCXX_3.4) [ISOCXX]
<code>basic_ios&lt;char, char_traits&lt;char&gt; &gt;::~~basic_ios()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::basic_ios<char, std::char_traits<char> >` specified in Table 9-237, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-237 libstdcxx - Class basic\_ios<char, char\_traits<char> > Data Interfaces**

typeid for basic_ios<char, char_traits<char> >(GLIBCXX_3.4) [CXXABI]
typeid name for basic_ios<char, char_traits<char> >(GLIBCXX_3.4) [CXXABI]
vtable for basic_ios<char, char_traits<char> >(GLIBCXX_3.4) [CXXABI]

**9.1.88 Class basic\_ios<wchar\_t, char\_traits<wchar\_t> >****9.1.88.1 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 9-238, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-238 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]



<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::exceptions(_Ios_Iostate)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::M_setstate(_Ios_Iostate)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::tie(basic_ostream&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::fill(wchar_t)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::init(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::clear(_Ios_Iostate)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::imbue(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::rdbuf(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::copyfmt(basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt; const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::setstate(_Ios_Iostate)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::basic_ios(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::basic_ios()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::basic_ios(basic_streambuf&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::basic_ios()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::~~basic_ios()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::~~basic_ios()(GLIBCXX_3.4) [ISOCXX]</code>
<code>basic_ios&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;::~~basic_ios()(GLIBCXX_3.4) [ISOCXX]</code>

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 9-239, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-239 libstdcxx - Class basic\_ios<wchar\_t, char\_traits<wchar\_t> > Data Interfaces**

typeinfo for basic_ios<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]
typeinfo name for basic_ios<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]
vtable for basic_ios<wchar_t, char_traits<wchar_t> >(GLIBCXX_3.4) [CXXABI]

**9.1.89 Class ios\_base::failure****9.1.89.1 Class data for ios\_base::failure**

The virtual table for the std::ios\_base::failure class is described by Table 9-240

**Table 9-240 Primary vtable for ios\_base::failure**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for ios_base::failure
vfunc[0]:	ios_base::failure::~~failure()
vfunc[1]:	ios_base::failure::~~failure()
vfunc[2]:	ios_base::failure::what() const

The Run Time Type Information for the std::ios\_base::failure class is described by Table 9-241

**Table 9-241 typeinfo for ios\_base::failure**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for ios_base::failure

**9.1.89.2 Interfaces for Class ios\_base::failure**

An LSB conforming implementation shall provide the generic methods for Class std::ios\_base::failure specified in Table 9-242, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-242 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 9-243, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-243 libstdcxx - Class ios\_base::failure Data Interfaces**

typeinfo for ios_base::failure(GLIBCXX_3.4) [CXXABI]
typeinfo name for ios_base::failure(GLIBCXX_3.4) [CXXABI]

vtable for ios_base::failure(GLIBCXX_3.4) [CXXABI]
--

## 9.1.90 Class `__timepunct<char>`

### 9.1.90.1 Class data for `__timepunct<char>`

The virtual table for the `std::__timepunct<char>` class is described by Table 9-244

**Table 9-244 Primary vtable for `__timepunct<char>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>__timepunct&lt;char&gt;</code>
vfunc[0]:	<code>__timepunct&lt;char&gt;::~~__timepunct()</code>
vfunc[1]:	<code>__timepunct&lt;char&gt;::~~__timepunct()</code>

The Run Time Type Information for the `std::__timepunct<char>` class is described by Table 9-245

**Table 9-245 typeid for `__timepunct<char>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>__timepunct&lt;char&gt;</code>

### 9.1.90.2 Interfaces for Class `__timepunct<char>`

An LSB conforming implementation shall provide the generic methods for Class `std::__timepunct<char>` specified in Table 9-246, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-246 libstdc++ - Class `__timepunct<char>` Function Interfaces**

<code>__timepunct&lt;char&gt;::_M_am_pm_format(char const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct&lt;char&gt;::_M_date_formats(char const**) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct&lt;char&gt;::_M_time_formats(char const**) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct&lt;char&gt;::_M_days_abbreviated(char const**) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct&lt;char&gt;::_M_date_time_formats(char const**) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct&lt;char&gt;::_M_months_abbreviated(char const**) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct&lt;char&gt;::_M_days(char const**) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct&lt;char&gt;::_M_am_pm(char const**) const(GLIBCXX_3.4) [ISOCXX]</code>

<code>__timepunct&lt;char&gt;::_M_months(char const**) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>__timepunct&lt;wchar_t&gt;::_M_am_pm_format(wchar_t const*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>__timepunct&lt;char&gt;::_M_initialize_timepunct(__locale_struct*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__timepunct&lt;char&gt;::~~__timepunct()</code> (GLIBCXX_3.4) [ISOCXX]
<code>__timepunct&lt;char&gt;::~~__timepunct()</code> (GLIBCXX_3.4) [ISOCXX]
<code>__timepunct&lt;char&gt;::~~__timepunct()</code> (GLIBCXX_3.4) [ISOCXX]
<code>bool has_facet&lt;__timepunct&lt;char&gt;&gt;(locale const&amp;)</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::__timepunct<char>` specified in Table 9-247, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-247 libstdcxx - Class `__timepunct<char>` Data Interfaces**

guard variable for <code>__timepunct&lt;char&gt;::id</code> (GLIBCXX_3.4) [CXXABI]
<code>__timepunct&lt;char&gt;::id</code> (GLIBCXX_3.4) [ISOCXX]
typeinfo for <code>__timepunct&lt;char&gt;</code> (GLIBCXX_3.4) [CXXABI]
typeinfo name for <code>__timepunct&lt;char&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>__timepunct&lt;char&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.91 Class `__timepunct<wchar_t>`

#### 9.1.91.1 Class data for `__timepunct<wchar_t>`

The virtual table for the `std::__timepunct<wchar_t>` class is described by Table 9-248

**Table 9-248 Primary vtable for `__timepunct<wchar_t>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>__timepunct&lt;wchar_t&gt;</code>
vfunc[0]:	<code>__timepunct&lt;wchar_t&gt;::~~__timepunct()</code>
vfunc[1]:	<code>__timepunct&lt;wchar_t&gt;::~~__timepunct()</code>

The Run Time Type Information for the `std::__timepunct<wchar_t>` class is described by Table 9-249

**Table 9-249 typeinfo for `__timepunct<wchar_t>`**

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

	__timepunct<wchar_t>
--	----------------------

### 9.1.91.2 Interfaces for Class \_\_timepunct<wchar\_t>

An LSB conforming implementation shall provide the generic methods for Class std::\_\_timepunct<wchar\_t> specified in Table 9-250, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-250 libstdcxx - Class \_\_timepunct<wchar\_t> Function Interfaces**

__timepunct<wchar_t>::_M_date_formats(wchar_t const**) const(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::_M_time_formats(wchar_t const**) const(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::_M_days_abbreviated(wchar_t const**) const(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::_M_date_time_formats(wchar_t const**) const(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::_M_months_abbreviated(wchar_t const**) const(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::_M_days(wchar_t const**) const(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::_M_am_pm(wchar_t const**) const(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::_M_months(wchar_t const**) const(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::_M_initialize_timepunct(__locale_struct*)(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::~~__timepunct()(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::~~__timepunct()(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::~~__timepunct()(GLIBCXX_3.4) [ISOCXX]
bool has_facet<__timepunct<wchar_t>>(locale const&)(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class std::\_\_timepunct<wchar\_t> specified in Table 9-251, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-251 libstdcxx - Class \_\_timepunct<wchar\_t> Data Interfaces**

guard variable for __timepunct<wchar_t>::id(GLIBCXX_3.4) [CXXABI]
__timepunct<wchar_t>::id(GLIBCXX_3.4) [ISOCXX]
typeid for __timepunct<wchar_t>(GLIBCXX_3.4) [CXXABI]
typeid name for __timepunct<wchar_t>(GLIBCXX_3.4) [CXXABI]
vtable for __timepunct<wchar_t>(GLIBCXX_3.4) [CXXABI]

## 9.1.92 Class `messages_base`

### 9.1.92.1 Class data for `messages_base`

The Run Time Type Information for the `std::messages_base` class is described by Table 9-252

**Table 9-252** `typeinfo` for `messages_base`

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

### 9.1.92.2 Interfaces for Class `messages_base`

No external methods are defined for `libstdc++` - Class `std::messages_base` in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for Class `std::messages_base` specified in Table 9-253, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-253** `libstdc++` - Class `messages_base` Data Interfaces

<code>typeinfo</code> for <code>messages_base</code> (GLIBCXX_3.4) [CXXABI]
<code>typeinfo</code> name for <code>messages_base</code> (GLIBCXX_3.4) [CXXABI]

## 9.1.93 Class `messages<char>`

### 9.1.93.1 Class data for `messages<char>`

The virtual table for the `std::messages<char>` class is described by Table 9-254

**Table 9-254** Primary vtable for `messages<char>`

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeinfo</code> for <code>messages&lt;char&gt;</code>
<code>vfunc</code> [0]:	<code>messages&lt;char&gt;::~~messages()</code>
<code>vfunc</code> [1]:	<code>messages&lt;char&gt;::~~messages()</code>
<code>vfunc</code> [2]:	<code>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</code>
<code>vfunc</code> [3]:	<code>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</code>
<code>vfunc</code> [4]:	<code>messages&lt;char&gt;::do_close(int) const</code>

### 9.1.93.2 Interfaces for Class `messages<char>`

An LSB conforming implementation shall provide the generic methods for Class `std::messages<char>` specified in Table 9-255, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-255 libstdcxx - Class `messages<char>` Function Interfaces**

<code>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]</code>
<code>messages&lt;char&gt;::_M_convert_from_char(char*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>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]</code>
<code>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]</code>
<code>messages&lt;char&gt;::open(basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt; const&amp;, locale const&amp;, char const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>messages&lt;char&gt;::close(int) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>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(GLIBCXX_3.4) [ISOCXX]</code>
<code>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(GLIBCXX_3.4) [ISOCXX]</code>
<code>messages&lt;char&gt;::do_close(int) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>messages&lt;char&gt;::~~messages()(GLIBCXX_3.4) [ISOCXX]</code>
<code>messages&lt;char&gt;::~~messages()(GLIBCXX_3.4) [ISOCXX]</code>
<code>messages&lt;char&gt;::~~messages()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::messages<char>` specified in Table 9-256, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-256 libstdcxx - Class `messages<char>` Data Interfaces**

guard variable for <code>messages&lt;char&gt;::id(GLIBCXX_3.4) [CXXABI]</code>
<code>messages&lt;char&gt;::id(GLIBCXX_3.4) [ISOCXX]</code>
typeinfo for <code>messages&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>
typeinfo name for <code>messages&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>
vtable for <code>messages&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.94 Class `messages<wchar_t>`

#### 9.1.94.1 Class data for `messages<wchar_t>`

The virtual table for the `std::messages<wchar_t>` class is described by Table 9-257

**Table 9-257 Primary vtable for messages<wchar\_t>**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for messages<wchar_t>
vfunc[0]:	messages<wchar_t>::~~messages()
vfunc[1]:	messages<wchar_t>::~~messages()
vfunc[2]:	messages<wchar_t>::do_open(basic_string<char, char_traits<char>, allocator<char> > const&, locale const&) const
vfunc[3]:	messages<wchar_t>::do_get(int, int, int, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&) const
vfunc[4]:	messages<wchar_t>::do_close(int) const

**9.1.94.2 Interfaces for Class messages<wchar\_t>**

An LSB conforming implementation shall provide the generic methods for Class std::messages<wchar\_t> specified in Table 9-258, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-258 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) [ISOCXX]
messages<wchar_t>::_M_convert_from_char(char*) const(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::get(int, int, int, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&) const(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::open(basic_string<char, char_traits<char>, allocator<char> > const&, locale const&) const(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::open(basic_string<char, char_traits<char>, allocator<char> > const&, locale const&, char const*) const(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::close(int) const(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::do_get(int, int, int, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&) const(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::do_open(basic_string<char, char_traits<char>, allocator<char> > const&, locale const&) const(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::do_close(int) const(GLIBCXX_3.4) [ISOCXX]



messages<wchar_t>::~messages()(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::~messages()(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::~messages()(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::messages<wchar_t>` specified in Table 9-259, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-259 libstdcxx - Class messages<wchar\_t> Data Interfaces**

guard variable for messages<wchar_t>::id(GLIBCXX_3.4) [CXXABI]
messages<wchar_t>::id(GLIBCXX_3.4) [ISOCXX]
typeid for messages<wchar_t>(GLIBCXX_3.4) [CXXABI]
typeid name for messages<wchar_t>(GLIBCXX_3.4) [CXXABI]
vtable for messages<wchar_t>(GLIBCXX_3.4) [CXXABI]

### 9.1.95 Class messages\_byname<char>

#### 9.1.95.1 Class data for messages\_byname<char>

The virtual table for the `std::messages_byname<char>` class is described by Table 9-260

**Table 9-260 Primary vtable for messages\_byname<char>**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for messages_byname<char>
vfunc[0]:	messages_byname<char>::~messages_byname()
vfunc[1]:	messages_byname<char>::~messages_byname()
vfunc[2]:	messages<char>::do_open(basic_string<char, char_traits<char>, allocator<char> > const&, locale const&) const
vfunc[3]:	messages<char>::do_get(int, int, int, basic_string<char, char_traits<char>, allocator<char> > const&) const
vfunc[4]:	messages<char>::do_close(int) const

The Run Time Type Information for the `std::messages_byname<char>` class is described by Table 9-261

**Table 9-261 typeid for messages\_byname<char>**

Base Vtable	vtable for
-------------	------------

	__cxxabiv1::__si_class_type_info
Name	typeid name for messages_byname<char>

### 9.1.95.2 Interfaces for Class messages\_byname<char>

An LSB conforming implementation shall provide the generic methods for Class std::messages\_byname<char> specified in Table 9-262, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-262 libstdc++ - 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 9-263, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-263 libstdc++ - Class messages\_byname<char> Data Interfaces**

typeid for messages_byname<char>(GLIBCXX_3.4) [CXXABI]
typeid name for messages_byname<char>(GLIBCXX_3.4) [CXXABI]
vtable for messages_byname<char>(GLIBCXX_3.4) [CXXABI]

### 9.1.96 Class messages\_byname<wchar\_t>

#### 9.1.96.1 Class data for messages\_byname<wchar\_t>

The virtual table for the std::messages\_byname<wchar\_t> class is described by Table 9-264

**Table 9-264 Primary vtable for messages\_byname<wchar\_t>**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for messages_byname<wchar_t>
vfunc[0]:	messages_byname<wchar_t>::~messages_byname()
vfunc[1]:	messages_byname<wchar_t>::~messages_byname()
vfunc[2]:	messages<wchar_t>::do_open(basic_string<char, char_traits<char>, allocator<char>> const&, locale const&) const
vfunc[3]:	messages<wchar_t>::do_get(int, int, int, basic_string<wchar_t, char_traits<wchar_t>,

	allocator<wchar_t> > const&) const
vfunc[4]:	messages<wchar_t>::do_close(int) const

The Run Time Type Information for the `std::messages_byname<wchar_t>` class is described by Table 9-265

**Table 9-265 typeid for messages\_byname<wchar\_t>**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>messages_byname&lt;wchar_t&gt;</code>

### 9.1.96.2 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 9-266, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-266 libstdc++ - Class `messages_byname<wchar_t>` Function Interfaces**

<code>messages_byname&lt;wchar_t&gt;::~messages_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>messages_byname&lt;wchar_t&gt;::~messages_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>messages_byname&lt;wchar_t&gt;::~messages_byname()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::messages_byname<wchar_t>` specified in Table 9-267, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-267 libstdc++ - Class `messages_byname<wchar_t>` Data Interfaces**

typeid for <code>messages_byname&lt;wchar_t&gt;</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>messages_byname&lt;wchar_t&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>messages_byname&lt;wchar_t&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.97 Class `num_punct<char>`

#### 9.1.97.1 Class data for `num_punct<char>`

The virtual table for the `std::num_punct<char>` class is described by Table 9-268

**Table 9-268 Primary vtable for `num_punct<char>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>num_punct&lt;char&gt;</code>
vfunc[0]:	<code>num_punct&lt;char&gt;::~num_punct()</code>
vfunc[1]:	<code>num_punct&lt;char&gt;::~num_punct()</code>
vfunc[2]:	<code>num_punct&lt;char&gt;::do_decimal_point(</code>

	) const
vfunc[3]:	numpunct<char>::do_thousands_sep ( ) const
vfunc[4]:	numpunct<char>::do_grouping() const
vfunc[5]:	numpunct<char>::do_truename() const
vfunc[6]:	numpunct<char>::do_falsename() const

The Run Time Type Information for the `std::numpunct<char>` class is described by Table 9-269

**Table 9-269 typeid for numpunct<char>**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>numpunct&lt;char&gt;</code>

### 9.1.97.2 Interfaces for Class `numpunct<char>`

An LSB conforming implementation shall provide the generic methods for Class `std::numpunct<char>` specified in Table 9-270, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-270 libstdcxx - Class `numpunct<char>` Function Interfaces**

<code>numpunct&lt;char&gt;::do_grouping() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::do_truename() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::do_falsename() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::decimal_point() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::thousands_sep() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::do_decimal_point() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::do_thousands_sep() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::grouping() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::truename() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::falsename() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::_M_initialize_numpunct(__locale_struct*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::~numpunct()(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::~numpunct()(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt;::~numpunct()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::num_punct<char>` specified in Table 9-271, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-271 libstdcxx - Class `num_punct<char>` Data Interfaces**

guard variable for <code>num_punct&lt;char&gt;::id(GLIBCXX_3.4)</code> [CXXABI]
<code>num_punct&lt;char&gt;::id(GLIBCXX_3.4)</code> [ISOCXX]
typeinfo for <code>num_punct&lt;char&gt;(GLIBCXX_3.4)</code> [CXXABI]
typeinfo name for <code>num_punct&lt;char&gt;(GLIBCXX_3.4)</code> [CXXABI]
vtable for <code>num_punct&lt;char&gt;(GLIBCXX_3.4)</code> [CXXABI]

## 9.1.98 Class `num_punct<wchar_t>`

### 9.1.98.1 Class data for `num_punct<wchar_t>`

The virtual table for the `std::num_punct<wchar_t>` class is described by Table 9-272

**Table 9-272 Primary vtable for `num_punct<wchar_t>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>num_punct&lt;wchar_t&gt;</code>
<code>vfunc[0]:</code>	<code>num_punct&lt;wchar_t&gt;::~~num_punct()</code>
<code>vfunc[1]:</code>	<code>num_punct&lt;wchar_t&gt;::~~num_punct()</code>
<code>vfunc[2]:</code>	<code>num_punct&lt;wchar_t&gt;::do_decimal_point() const</code>
<code>vfunc[3]:</code>	<code>num_punct&lt;wchar_t&gt;::do_thousands_sep() const</code>
<code>vfunc[4]:</code>	<code>num_punct&lt;wchar_t&gt;::do_grouping() const</code>
<code>vfunc[5]:</code>	<code>num_punct&lt;wchar_t&gt;::do_truename() const</code>
<code>vfunc[6]:</code>	<code>num_punct&lt;wchar_t&gt;::do_falsename() const</code>

The Run Time Type Information for the `std::num_punct<wchar_t>` class is described by Table 9-273

**Table 9-273 typeinfo for `num_punct<wchar_t>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>num_punct&lt;wchar_t&gt;</code>

### 9.1.98.2 Interfaces for Class `numpunct<wchar_t>`

An LSB conforming implementation shall provide the generic methods for Class `std::numpunct<wchar_t>` specified in Table 9-274, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-274 libstdcxx - Class `numpunct<wchar_t>` Function Interfaces**

<code>numpunct&lt;wchar_t&gt;::do_grouping() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::do_truename() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::do_falsename() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::decimal_point() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::thousands_sep() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::do_decimal_point() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::do_thousands_sep() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::grouping() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::truename() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::falsename() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::_M_initialize_numpunct(__locale_struct*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::~~numpunct()(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::~~numpunct()(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt;::~~numpunct()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::numpunct<wchar_t>` specified in Table 9-275, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-275 libstdcxx - Class `numpunct<wchar_t>` Data Interfaces**

guard variable for <code>numpunct&lt;wchar_t&gt;::id(GLIBCXX_3.4) [CXXABI]</code>
<code>numpunct&lt;wchar_t&gt;::id(GLIBCXX_3.4) [ISOCXX]</code>
typeinfo for <code>numpunct&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
typeinfo name for <code>numpunct&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
vtable for <code>numpunct&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.99 Class `numpunct_byname<char>`

#### 9.1.99.1 Class data for `numpunct_byname<char>`

The virtual table for the `std::numpunct_byname<char>` class is described by Table 9-276

**Table 9-276 Primary vtable for `numpunct_byname<char>`**

Base Offset	0
-------------	---

Virtual Base Offset	0
RTTI	typeid for numpunct_byname<char>
vfunc[0]:	numpunct_byname<char>::~~numpu nct_byname()
vfunc[1]:	numpunct_byname<char>::~~numpu nct_byname()
vfunc[2]:	numpunct<char>::do_decimal_point( ) const
vfunc[3]:	numpunct<char>::do_thousands_sep ( ) const
vfunc[4]:	numpunct<char>::do_grouping() const
vfunc[5]:	numpunct<char>::do_truename() const
vfunc[6]:	numpunct<char>::do_falsename() const

The Run Time Type Information for the `std::numpunct_byname<char>` class is described by Table 9-277

**Table 9-277 typeid for numpunct\_byname<char>**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for numpunct_byname<char>

### 9.1.99.2 Interfaces for Class numpunct\_byname<char>

An LSB conforming implementation shall provide the generic methods for Class `std::numpunct_byname<char>` specified in Table 9-278, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-278 libstdc++ - Class numpunct\_byname<char> Function Interfaces**

<code>numpunct_byname&lt;char&gt;::~~numpunct_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct_byname&lt;char&gt;::~~numpunct_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct_byname&lt;char&gt;::~~numpunct_byname()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::numpunct_byname<char>` specified in Table 9-279, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-279 libstdc++ - Class numpunct\_byname<char> Data Interfaces**

<code>typeid for numpunct_byname&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for numpunct_byname&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>

vtable for <code>numpunct_byname&lt;char&gt;(GLIBCXX_3.4)</code> [CXXABI]
---

### 9.1.100 Class `numpunct_byname<wchar_t>`

#### 9.1.100.1 Class data for `numpunct_byname<wchar_t>`

The virtual table for the `std::numpunct_byname<wchar_t>` class is described by Table 9-280

**Table 9-280 Primary vtable for `numpunct_byname<wchar_t>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>numpunct_byname&lt;wchar_t&gt;</code>
<code>vfunc[0]:</code>	<code>numpunct_byname&lt;wchar_t&gt;::~~numpunct_byname()</code>
<code>vfunc[1]:</code>	<code>numpunct_byname&lt;wchar_t&gt;::~~numpunct_byname()</code>
<code>vfunc[2]:</code>	<code>numpunct&lt;wchar_t&gt;::do_decimal_point() const</code>
<code>vfunc[3]:</code>	<code>numpunct&lt;wchar_t&gt;::do_thousands_sep() const</code>
<code>vfunc[4]:</code>	<code>numpunct&lt;wchar_t&gt;::do_grouping() const</code>
<code>vfunc[5]:</code>	<code>numpunct&lt;wchar_t&gt;::do_truename() const</code>
<code>vfunc[6]:</code>	<code>numpunct&lt;wchar_t&gt;::do_falsename() const</code>

The Run Time Type Information for the `std::numpunct_byname<wchar_t>` class is described by Table 9-281

**Table 9-281 typeinfo for `numpunct_byname<wchar_t>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>numpunct_byname&lt;wchar_t&gt;</code>

#### 9.1.100.2 Interfaces for Class `numpunct_byname<wchar_t>`

An LSB conforming implementation shall provide the generic methods for Class `std::numpunct_byname<wchar_t>` specified in Table 9-282, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-282 `libstdcxx` - Class `numpunct_byname<wchar_t>` Function Interfaces**

<code>numpunct_byname&lt;wchar_t&gt;::~~numpunct_byname()(GLIBCXX_3.4)</code>
---



[ISOCXX]
numpunct_byname<wchar_t>::~numpunct_byname()(GLIBCXX_3.4) [ISOCXX]
numpunct_byname<wchar_t>::~numpunct_byname()(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::numpunct_byname<wchar_t>` specified in Table 9-283, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-283 libstdcxx - Class `numpunct_byname<wchar_t>` Data Interfaces**

typeid for <code>numpunct_byname&lt;wchar_t&gt;</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>numpunct_byname&lt;wchar_t&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>numpunct_byname&lt;wchar_t&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.101 Class `__codecvt_abstract_base<char, char, __mbstate_t>`

#### 9.1.101.1 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 ISO/IEC 23360.

### 9.1.102 Class `__codecvt_abstract_base<wchar_t, char, __mbstate_t>`

#### 9.1.102.1 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 9-284

**Table 9-284 Primary vtable for `__codecvt_abstract_base<wchar_t, char, __mbstate_t>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>__codecvt_abstract_base&lt;wchar_t, char, __mbstate_t&gt;</code>
vfunc[0]:	
vfunc[1]:	
vfunc[2]:	<code>__cxa_pure_virtual</code>
vfunc[3]:	<code>__cxa_pure_virtual</code>
vfunc[4]:	<code>__cxa_pure_virtual</code>
vfunc[5]:	<code>__cxa_pure_virtual</code>

vfunc[6]:	__cxa_pure_virtual
vfunc[7]:	__cxa_pure_virtual
vfunc[8]:	__cxa_pure_virtual

### 9.1.102.2 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for Class `std::__codecvt_abstract_base<wchar_t, char, __mbstate_t>` specified in Table 9-285, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-285 `libstdcxx` - Class `__codecvt_abstract_base<wchar_t, char, __mbstate_t>` Data Interfaces**

typeinfo for <code>__codecvt_abstract_base&lt;wchar_t, char, __mbstate_t&gt;</code> (GLIBCXX_3.4) [CXXABI]
typeinfo name for <code>__codecvt_abstract_base&lt;wchar_t, char, __mbstate_t&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>__codecvt_abstract_base&lt;wchar_t, char, __mbstate_t&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.103 Class `codecvt_base`

#### 9.1.103.1 Class data for `codecvt_base`

The Run Time Type Information for the `std::codecvt_base` class is described by Table 9-286

**Table 9-286 typeinfo for `codecvt_base`**

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

#### 9.1.103.2 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for Class `std::codecvt_base` specified in Table 9-287, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-287 `libstdcxx` - Class `codecvt_base` Data Interfaces**

typeinfo for <code>codecvt_base</code> (GLIBCXX_3.4) [CXXABI]
typeinfo name for <code>codecvt_base</code> (GLIBCXX_3.4) [CXXABI]

**9.1.104 Class codecvt<char, char, \_\_mbstate\_t>****9.1.104.1 Class data for codecvt<char, char, \_\_mbstate\_t>**

The virtual table for the `std::codecvt<char, char, __mbstate_t>` class is described by Table 9-288

**Table 9-288 Primary vtable for codecvt<char, char, \_\_mbstate\_t>**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>codecvt&lt;char, char, __mbstate_t&gt;</code>
vfunc[0]:	<code>codecvt&lt;char, char, __mbstate_t&gt;::~~codecvt()</code>
vfunc[1]:	<code>codecvt&lt;char, char, __mbstate_t&gt;::~~codecvt()</code>
vfunc[2]:	<code>codecvt&lt;char, char, __mbstate_t&gt;::do_out(__mbstate_t&amp;, char const*, char const*, char const*&amp;, char*, char*, char*&amp;) const</code>
vfunc[3]:	<code>codecvt&lt;char, char, __mbstate_t&gt;::do_unshift(__mbstate_t&amp;, char*, char*, char*&amp;) const</code>
vfunc[4]:	<code>codecvt&lt;char, char, __mbstate_t&gt;::do_in(__mbstate_t&amp;, char const*, char const*, char const*&amp;, char*, char*, char*&amp;) const</code>
vfunc[5]:	<code>codecvt&lt;char, char, __mbstate_t&gt;::do_encoding() const</code>
vfunc[6]:	<code>codecvt&lt;char, char, __mbstate_t&gt;::do_always_noconv() const</code>
vfunc[7]:	See The Architecture Specific Specification
vfunc[8]:	<code>codecvt&lt;char, char, __mbstate_t&gt;::do_max_length() const</code>

The Run Time Type Information for the `std::codecvt<char, char, __mbstate_t>` class is described by Table 9-289

**Table 9-289 typeinfo for codecvt<char, char, \_\_mbstate\_t>**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>codecvt&lt;char, char, __mbstate_t&gt;</code>

### 9.1.104.2 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 9-290

**Table 9-290 Primary vtable for `__codecvt_abstract_base<char, char, __mbstate_t>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>__codecvt_abstract_base&lt;char, char, __mbstate_t&gt;</code>
vfunc[0]:	
vfunc[1]:	
vfunc[2]:	<code>__cxa_pure_virtual</code>
vfunc[3]:	<code>__cxa_pure_virtual</code>
vfunc[4]:	<code>__cxa_pure_virtual</code>
vfunc[5]:	<code>__cxa_pure_virtual</code>
vfunc[6]:	<code>__cxa_pure_virtual</code>
vfunc[7]:	<code>__cxa_pure_virtual</code>
vfunc[8]:	<code>__cxa_pure_virtual</code>

### 9.1.104.3 Interfaces for 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 9-291, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-291 `libstdcxx` - Class `codecvt<char, char, __mbstate_t>` Function Interfaces**

<code>codecvt&lt;char, char, __mbstate_t&gt;::do_unshift(__mbstate_t&amp;, char*, char*, char*&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt&lt;char, char, __mbstate_t&gt;::do_encoding() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt&lt;char, char, __mbstate_t&gt;::do_max_length() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt&lt;char, char, __mbstate_t&gt;::do_always_noconv() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt&lt;char, char, __mbstate_t&gt;::do_in(__mbstate_t&amp;, char const*, char const*, char const*&amp;, char*, char*, char*&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt&lt;char, char, __mbstate_t&gt;::do_out(__mbstate_t&amp;, char const*, char const*, char const*&amp;, char*, char*, char*&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt&lt;char, char, __mbstate_t&gt;::~~codecvt()(GLIBCXX_3.4) [ISOCXX]</code>

<code>codecvt&lt;char, char, __mbstate_t&gt;::~codecvt()(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt&lt;char, char, __mbstate_t&gt;::~codecvt()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::codecvt<char, char, __mbstate_t>` specified in Table 9-292, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-292 libstdcxx - Class `codecvt<char, char, __mbstate_t>` Data Interfaces**

<code>codecvt&lt;char, char, __mbstate_t&gt;::id(GLIBCXX_3.4) [ISOCXX]</code>
<code>typeid for __codecvt_abstract_base&lt;char, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid for codecvt&lt;char, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for __codecvt_abstract_base&lt;char, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for codecvt&lt;char, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for __codecvt_abstract_base&lt;char, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for codecvt&lt;char, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.105 Class `codecvt<wchar_t, char, __mbstate_t>`

#### 9.1.105.1 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 9-293

**Table 9-293 Primary vtable for `codecvt<wchar_t, char, __mbstate_t>`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid for codecvt&lt;wchar_t, char, __mbstate_t&gt;</code>
<code>vfunc[0]:</code>	<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::~codecvt()</code>
<code>vfunc[1]:</code>	<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::~codecvt()</code>
<code>vfunc[2]:</code>	<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_out(__mbstate_t&amp;, wchar_t const*, wchar_t const*, wchar_t const*&amp;, char*, char*, char*&amp;) const</code>
<code>vfunc[3]:</code>	<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_unshift(__mbstate_t&amp;, char*, char*, char*&amp;) const</code>

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 The Architecture Specific Specification
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 9-294

**Table 9-294** `typeinfo` for `codecvt<wchar_t, char, __mbstate_t>`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	<code>typeinfo</code> name for <code>codecvt&lt;wchar_t, char, __mbstate_t&gt;</code>

### 9.1.105.2 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 9-295, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-295** `libstdcxx` - Class `codecvt<wchar_t, char, __mbstate_t>` Function Interfaces

<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_unshift(__mbstate_t&amp;, char*, char*, char*&amp;) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_encoding() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_max_length() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_always_noconv() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_in(__mbstate_t&amp;, char const*, char const*, char const*&amp;, wchar_t*, wchar_t*, wchar_t*&amp;) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_out(__mbstate_t&amp;, wchar_t const*, wchar_t const*, wchar_t const*&amp;, char*, char*, char*&amp;) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::~codecvt()</code> (GLIBCXX_3.4) [ISOCXX]

<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::~~codecvt()(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::~~codecvt()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::codecvt<wchar_t, char, __mbstate_t>` specified in Table 9-296, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-296 libstdcxx - Class `codecvt<wchar_t, char, __mbstate_t>` Data Interfaces**

<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::id(GLIBCXX_3.4) [ISOCXX]</code>
<code>typeid for codecvt&lt;wchar_t, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for codecvt&lt;wchar_t, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for codecvt&lt;wchar_t, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.106 Class `codecvt_byname<char, char, __mbstate_t>`

#### 9.1.106.1 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 9-297

**Table 9-297 Primary vtable for `codecvt_byname<char, char, __mbstate_t>`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid for codecvt_byname&lt;char, char, __mbstate_t&gt;</code>
<code>vfunc[0]:</code>	<code>codecvt_byname&lt;char, char, __mbstate_t&gt;::~~codecvt_byname()</code>
<code>vfunc[1]:</code>	<code>codecvt_byname&lt;char, char, __mbstate_t&gt;::~~codecvt_byname()</code>
<code>vfunc[2]:</code>	<code>codecvt&lt;char, char, __mbstate_t&gt;::do_out(__mbstate_t&amp;, char const*, char const*, char const*&amp;, char*, char*, char*&amp;) const</code>
<code>vfunc[3]:</code>	<code>codecvt&lt;char, char, __mbstate_t&gt;::do_unshift(__mbstate_t&amp;, char*, char*, char*&amp;) const</code>
<code>vfunc[4]:</code>	<code>codecvt&lt;char, char, __mbstate_t&gt;::do_in(__mbstate_t&amp;, char const*, char const*, char const*&amp;, char*, char*, char*&amp;) const</code>
<code>vfunc[5]:</code>	<code>codecvt&lt;char, char, __mbstate_t&gt;::do_encoding() const</code>
<code>vfunc[6]:</code>	<code>codecvt&lt;char, char,</code>

	<code>__mbstate_t::do_always_noconv()</code> const
<code>vfunc[7]:</code>	See The Architecture Specific Specification
<code>vfunc[8]:</code>	<code>codecvt&lt;char, char, __mbstate_t::do_max_length()</code> const

The Run Time Type Information for the `std::codecvt_byname<char, char, __mbstate_t>` class is described by Table 9-298

**Table 9-298 typeid for `codecvt_byname<char, char, __mbstate_t>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>codecvt_byname&lt;char, char, __mbstate_t&gt;</code>

### 9.1.106.2 Interfaces for Class `codecvt_byname<char, char, __mbstate_t>`

An LSB conforming implementation shall provide the generic methods for Class `std::codecvt_byname<char, char, __mbstate_t>` specified in Table 9-299, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-299 libstdc++ - Class `codecvt_byname<char, char, __mbstate_t>` Function Interfaces**

<code>codecvt_byname&lt;char, char, __mbstate_t::~codecvt_byname&gt;(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt_byname&lt;char, char, __mbstate_t::~codecvt_byname&gt;(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt_byname&lt;char, char, __mbstate_t::~codecvt_byname&gt;(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::codecvt_byname<char, char, __mbstate_t>` specified in Table 9-300, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-300 libstdc++ - Class `codecvt_byname<char, char, __mbstate_t>` Data Interfaces**

typeid for <code>codecvt_byname&lt;char, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
typeid name for <code>codecvt_byname&lt;char, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
vtable for <code>codecvt_byname&lt;char, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>



**9.1.107 Class codecvt\_byname<wchar\_t, char, \_\_mbstate\_t>****9.1.107.1 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 9-301

**Table 9-301 Primary vtable for codecvt\_byname<wchar\_t, char, \_\_mbstate\_t>**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>codecvt_byname&lt;wchar_t, char, __mbstate_t&gt;</code>
vfunc[0]:	<code>codecvt_byname&lt;wchar_t, char, __mbstate_t&gt;::~~codecvt_byname()</code>
vfunc[1]:	<code>codecvt_byname&lt;wchar_t, char, __mbstate_t&gt;::~~codecvt_byname()</code>
vfunc[2]:	<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_out(__mbstate_t&amp;, wchar_t const*, wchar_t const*, wchar_t const*&amp;, char*, char*, char*&amp;) const</code>
vfunc[3]:	<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_unshift(__mbstate_t&amp;, char*, char*, char*&amp;) const</code>
vfunc[4]:	<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_in(__mbstate_t&amp;, char const*, char const*, char const*&amp;, wchar_t*, wchar_t*, wchar_t*&amp;) const</code>
vfunc[5]:	<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_encoding() const</code>
vfunc[6]:	<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_always_noconv() const</code>
vfunc[7]:	See The Architecture Specific Specification
vfunc[8]:	<code>codecvt&lt;wchar_t, char, __mbstate_t&gt;::do_max_length() const</code>

The Run Time Type Information for the `std::codecvt_byname<wchar_t, char, __mbstate_t>` class is described by Table 9-302

**Table 9-302 typeid for codecvt\_byname<wchar\_t, char, \_\_mbstate\_t>**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
-------------	--

Name	typeid name for codecvt_byname<wchar_t, char, __mbstate_t>
------	--

### 9.1.107.2 Class data for collate\_byname<wchar\_t>

The virtual table for the `std::collate_byname<wchar_t>` class is described by Table 9-303

**Table 9-303 Primary vtable for collate\_byname<wchar\_t>**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for collate_byname<wchar_t>
vfunc[0]:	collate_byname<wchar_t>::~~collate_byname()
vfunc[1]:	collate_byname<wchar_t>::~~collate_byname()
vfunc[2]:	collate<wchar_t>::do_compare(wchar_t const*, wchar_t const*, wchar_t const*, wchar_t const*) const
vfunc[3]:	collate<wchar_t>::do_transform(wchar_t const*, wchar_t const*) const
vfunc[4]:	collate<wchar_t>::do_hash(wchar_t const*, wchar_t const*) const

The Run Time Type Information for the `std::collate_byname<wchar_t>` class is described by Table 9-304

**Table 9-304 typeid for collate\_byname<wchar\_t>**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for collate_byname<wchar_t>

### 9.1.107.3 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 9-305, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-305 libstdc++ - Class codecvt\_byname<wchar\_t, char, \_\_mbstate\_t> Function Interfaces**

codecvt_byname<wchar_t, char, __mbstate_t>::~~codecvt_byname()(GLIBCXX_3.4) [ISOCXX]
codecvt_byname<wchar_t, char,

<code>__mbstate_t&gt;::~codecvt_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt_byname&lt;wchar_t, char, __mbstate_t&gt;::~codecvt_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate_byname&lt;wchar_t&gt;::~collate_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate_byname&lt;wchar_t&gt;::~collate_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate_byname&lt;wchar_t&gt;::~collate_byname()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::codecvt_byname<wchar_t, char, __mbstate_t>` specified in Table 9-306, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-306 libstdcxx - Class `codecvt_byname<wchar_t, char, __mbstate_t>` Data Interfaces**

<code>typeid for codecvt_byname&lt;wchar_t, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid for collate_byname&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for codecvt_byname&lt;wchar_t, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for collate_byname&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for codecvt_byname&lt;wchar_t, char, __mbstate_t&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for collate_byname&lt;wchar_t&gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.108 Class `collate<char>`

#### 9.1.108.1 Class data for `collate<char>`

The virtual table for the `std::collate<char>` class is described by Table 9-307

**Table 9-307 Primary vtable for `collate<char>`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid for collate&lt;char&gt;</code>
<code>vfunc[0]:</code>	<code>collate&lt;char&gt;::~~collate()</code>
<code>vfunc[1]:</code>	<code>collate&lt;char&gt;::~~collate()</code>
<code>vfunc[2]:</code>	<code>collate&lt;char&gt;::do_compare(char const*, char const*, char const*, char const*) const</code>
<code>vfunc[3]:</code>	<code>collate&lt;char&gt;::do_transform(char const*, char const*) const</code>
<code>vfunc[4]:</code>	<code>collate&lt;char&gt;::do_hash(char const*, char const*) const</code>

The Run Time Type Information for the `std::collate<char>` class is described by Table 9-308

**Table 9-308 typeid for collate<char>**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>collate&lt;char&gt;</code>

### 9.1.108.2 Interfaces for Class `collate<char>`

An LSB conforming implementation shall provide the generic methods for Class `std::collate<char>` specified in Table 9-309, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-309 libstdcxx - Class `collate<char>` Function Interfaces**

<code>collate&lt;char&gt;::_M_compare(char const*, char const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate&lt;char&gt;::_do_compare(char const*, char const*, char const*, char const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate&lt;char&gt;::_do_transform(char const*, char const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate&lt;char&gt;::_hash(char const*, char const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate&lt;char&gt;::_compare(char const*, char const*, char const*, char const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate&lt;char&gt;::_do_hash(char const*, char const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate&lt;char&gt;::_transform(char const*, char const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate&lt;char&gt;::~~collate()(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate&lt;char&gt;::~~collate()(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate&lt;char&gt;::~~collate()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::collate<char>` specified in Table 9-310, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-310 libstdcxx - Class `collate<char>` Data Interfaces**

guard variable for <code>collate&lt;char&gt;::_id(GLIBCXX_3.4) [CXXABI]</code>
<code>collate&lt;char&gt;::_id(GLIBCXX_3.4) [ISOCXX]</code>
typeid for <code>collate&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>
typeid name for <code>collate&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>
vtable for <code>collate&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.109 Class `collate<wchar_t>`

#### 9.1.109.1 Class data for `collate<wchar_t>`

The virtual table for the `std::collate<wchar_t>` class is described by Table 9-311

**Table 9-311 Primary vtable for `collate<wchar_t>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>collate&lt;wchar_t&gt;</code>
<code>vfunc[0]:</code>	<code>collate&lt;wchar_t&gt;::~~collate()</code>
<code>vfunc[1]:</code>	<code>collate&lt;wchar_t&gt;::~~collate()</code>
<code>vfunc[2]:</code>	<code>collate&lt;wchar_t&gt;::do_compare(wchar_t const*, wchar_t const*, wchar_t const*, wchar_t const*) const</code>
<code>vfunc[3]:</code>	<code>collate&lt;wchar_t&gt;::do_transform(wchar_t const*, wchar_t const*) const</code>
<code>vfunc[4]:</code>	<code>collate&lt;wchar_t&gt;::do_hash(wchar_t const*, wchar_t const*) const</code>

The Run Time Type Information for the `std::collate<wchar_t>` class is described by Table 9-312

**Table 9-312 typeinfo for `collate<wchar_t>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>collate&lt;wchar_t&gt;</code>

#### 9.1.109.2 Interfaces for Class `collate<wchar_t>`

An LSB conforming implementation shall provide the generic methods for Class `std::collate<wchar_t>` specified in Table 9-313, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-313 `libstdcxx` - Class `collate<wchar_t>` Function Interfaces**

<code>collate&lt;wchar_t&gt;::_M_compare(wchar_t const*, wchar_t const*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>collate&lt;wchar_t&gt;::do_compare(wchar_t const*, wchar_t const*, wchar_t const*, wchar_t const*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>collate&lt;wchar_t&gt;::do_transform(wchar_t const*, wchar_t const*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>collate&lt;wchar_t&gt;::hash(wchar_t const*, wchar_t const*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>collate&lt;wchar_t&gt;::compare(wchar_t const*, wchar_t const*, wchar_t const*, wchar_t const*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>collate&lt;wchar_t&gt;::do_hash(wchar_t const*, wchar_t const*)</code>

const(GLIBCXX_3.4) [ISOCXX]
collate<wchar_t>::transform(wchar_t const*, wchar_t const*) const(GLIBCXX_3.4) [ISOCXX]
collate<wchar_t>::~collate()(GLIBCXX_3.4) [ISOCXX]
collate<wchar_t>::~collate()(GLIBCXX_3.4) [ISOCXX]
collate<wchar_t>::~collate()(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::collate<wchar_t>` specified in Table 9-314, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-314 libstdc++ - Class `collate<wchar_t>` Data Interfaces**

guard variable for <code>collate&lt;wchar_t&gt;::id(GLIBCXX_3.4)</code> [CXXABI]
<code>collate&lt;wchar_t&gt;::id(GLIBCXX_3.4)</code> [ISOCXX]
typeinfo for <code>collate&lt;wchar_t&gt;(GLIBCXX_3.4)</code> [CXXABI]
typeinfo name for <code>collate&lt;wchar_t&gt;(GLIBCXX_3.4)</code> [CXXABI]
vtable for <code>collate&lt;wchar_t&gt;(GLIBCXX_3.4)</code> [CXXABI]

### 9.1.110 Class `collate_byname<char>`

#### 9.1.110.1 Class data for `collate_byname<char>`

The virtual table for the `std::collate_byname<char>` class is described by Table 9-315

**Table 9-315 Primary vtable for `collate_byname<char>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>collate_byname&lt;char&gt;</code>
vfunc[0]:	<code>collate_byname&lt;char&gt;::~collate_byname()</code>
vfunc[1]:	<code>collate_byname&lt;char&gt;::~collate_byname()</code>
vfunc[2]:	<code>collate&lt;char&gt;::do_compare(char const*, char const*, char const*, char const*) const</code>
vfunc[3]:	<code>collate&lt;char&gt;::do_transform(char const*, char const*) const</code>
vfunc[4]:	<code>collate&lt;char&gt;::do_hash(char const*, char const*) const</code>

The Run Time Type Information for the `std::collate_byname<char>` class is described by Table 9-316

**Table 9-316 typeid for collate\_byname<char>**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for collate_byname<char>

**9.1.110.2 Interfaces for Class collate\_byname<char>**

An LSB conforming implementation shall provide the generic methods for Class `std::collate_byname<char>` specified in Table 9-317, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-317 libstdcxx - Class collate\_byname<char> Function Interfaces**

<code>collate_byname&lt;char&gt;::~collate_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate_byname&lt;char&gt;::~collate_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate_byname&lt;char&gt;::~collate_byname()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::collate_byname<char>` specified in Table 9-318, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-318 libstdcxx - Class collate\_byname<char> Data Interfaces**

<code>typeid for collate_byname&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for collate_byname&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for collate_byname&lt;char&gt;(GLIBCXX_3.4) [CXXABI]</code>

**9.1.111 Class collate\_byname<wchar\_t>****9.1.111.1 Interfaces for Class collate\_byname<wchar\_t>**

No external methods are defined for `libstdcxx - Class std::collate_byname<wchar_t>` in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

**9.1.112 Class time\_base****9.1.112.1 Class data for time\_base**

The Run Time Type Information for the `std::time_base` class is described by Table 9-319

**Table 9-319 typeid for time\_base**

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeid name for <code>time_base</code>

**9.1.112.2 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 ISO/IEC 23360.

An LSB conforming implementation shall provide the generic data interfaces for Class `std::time_base` specified in Table 9-320, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-320 libstdcxx - Class `time_base` Data Interfaces**

typeinfo for <code>time_base(GLIBCXX_3.4)</code> [CXXABI]
typeinfo name for <code>time_base(GLIBCXX_3.4)</code> [CXXABI]

### 9.1.113 Class `time_get_byname<char, istreambuf_iterator<char, char_traits<char>>>`

#### 9.1.113.1 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 9-321

**Table 9-321 Primary vtable for `time_get_byname<char, istreambuf_iterator<char, char_traits<char>>>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code>
<code>vfunc[0]:</code>	<code>time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::~time_get_byname()</code>
<code>vfunc[1]:</code>	<code>time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::~time_get_byname()</code>
<code>vfunc[2]:</code>	<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::do_date_order() const</code>
<code>vfunc[3]:</code>	<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::do_get_time(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;, tm*) const</code>
<code>vfunc[4]:</code>	<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::do_get_date(istreambuf_iterator&lt;c</code>



	<pre> char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _ios_istate&amp;, tm*) const </pre>
vfunc[5]:	<pre> time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_get_weekday(istreambuf_iterat or&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _ios_istate&amp;, tm*) const </pre>
vfunc[6]:	<pre> time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_get_monthname(istreambuf_it erator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _ios_istate&amp;, tm*) const </pre>
vfunc[7]:	<pre> time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_get_year(istreambuf_iterator&lt;c har, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _ios_istate&amp;, tm*) const </pre>

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 9-322

**Table 9-322** typeinfo for `time_get_byname<char, istreambuf_iterator<char, char_traits<char> > >`

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>time_get_byname&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</code>

### 9.1.113.2 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 9-323, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-323 libstdcxx - Class time\_get\_byname<char, istreambuf\_iterator<char, char\_traits<char>>> Function Interfaces**

time_get_byname<char, istreambuf_iterator<char, char_traits<char>>> >::~time_get_byname()(GLIBCXX_3.4) [ISOCXX]
time_get_byname<char, istreambuf_iterator<char, char_traits<char>>> >::~time_get_byname()(GLIBCXX_3.4) [ISOCXX]
time_get_byname<char, istreambuf_iterator<char, char_traits<char>>> >::~time_get_byname()(GLIBCXX_3.4) [ISOCXX]

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 9-324, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-324 libstdcxx - Class time\_get\_byname<char, istreambuf\_iterator<char, char\_traits<char>>> Data Interfaces**

typeid for time_get_byname<char, istreambuf_iterator<char, char_traits<char>>>(GLIBCXX_3.4) [CXXABI]
typeid name for time_get_byname<char, istreambuf_iterator<char, char_traits<char>>>(GLIBCXX_3.4) [CXXABI]
vtable for time_get_byname<char, istreambuf_iterator<char, char_traits<char>>>(GLIBCXX_3.4) [CXXABI]

### 9.1.114 Class time\_get\_byname<wchar\_t, istreambuf\_iterator<wchar\_t, char\_traits<wchar\_t>>>

#### 9.1.114.1 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 9-325

**Table 9-325 Primary vtable for time\_get\_byname<wchar\_t, istreambuf\_iterator<wchar\_t, char\_traits<wchar\_t>>>**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>
vfunc[0]:	time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>> >::~time_get_byname()
vfunc[1]:	time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>

	>::~time_get_byname()
vfunc[2]:	time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::do_date_order() const
vfunc[3]:	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_iterat or<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_it erator<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

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 9-326

**Table 9-326 typeinfo for time\_get\_byname<wchar\_t, istreambuf\_iterator<wchar\_t, char\_traits<wchar\_t>>>**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>

### 9.1.114.2 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, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>` specified in Table 9-327, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-327 libstdcxx - Class time\_get\_byname<wchar\_t, istreambuf\_iterator<wchar\_t, char\_traits<wchar\_t>>> Function Interfaces**

<code>time_get_byname&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::~time_get_byname()</code> (GLIBCXX_3.4) [ISOCXX]
<code>time_get_byname&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::~time_get_byname()</code> (GLIBCXX_3.4) [ISOCXX]
<code>time_get_byname&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::~time_get_byname()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>` specified in Table 9-328, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-328 libstdcxx - Class time\_get\_byname<wchar\_t, istreambuf\_iterator<wchar\_t, char\_traits<wchar\_t>>> Data Interfaces**

typeinfo for <code>time_get_byname&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
typeinfo name for <code>time_get_byname&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>time_get_byname&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.115 Class time\_put\_byname<char, ostreambuf\_iterator<char, char\_traits<char>>>

#### 9.1.115.1 Class data for time\_put\_byname<char, ostreambuf\_iterator<char, char\_traits<char>>>

The virtual table for the `std::time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>` class is described by Table 9-329

**Table 9-329 Primary vtable for `time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid</code> for <code>time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code>
<code>vfunc[0]:</code>	<code>time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::~time_put_byname()</code>
<code>vfunc[1]:</code>	<code>time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::~time_put_byname()</code>
<code>vfunc[2]:</code>	<code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::do_put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, ios_base&amp;, char, tm const*, char, char) const</code>

The Run Time Type Information for the `std::time_put_byname<char, std::ostreambuf_iterator<char, std::char_traits<char>>>` class is described by Table 9-330

**Table 9-330 `typeid` for `time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	<code>typeid</code> name for <code>time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code>

### 9.1.115.2 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 9-331, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-331 `libstdcxx` - Class `time_put_byname<char, ostreambuf_iterator<char, char_traits<char>>>` Function Interfaces**

<code>time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::~time_put_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::~time_put_byname()(GLIBCXX_3.4) [ISOCXX]</code>

<pre>time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;::~time_put_byname()(GLIBCXX_3.4) [ISOCXX]</pre>
---

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 9-332, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-332 libstdcxx - Class `time_put_byname<char, ostreambuf_iterator<char, char_traits<char> > >` Data Interfaces**

<pre>typeid for time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</pre>
<pre>typeid name for time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</pre>
<pre>vtable for time_put_byname&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</pre>

### 9.1.116 Class `time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >`

#### 9.1.116.1 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 9-333

**Table 9-333 Primary vtable for `time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >`**

Base Offset	0
Virtual Base Offset	0
RTTI	<pre>typeid for time_put_byname&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</pre>
vfunc[0]:	<pre>time_put_byname&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;::~time_put_byname()</pre>
vfunc[1]:	<pre>time_put_byname&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;::~time_put_byname()</pre>
vfunc[2]:	<pre>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;::do_put(ostreambuf_iterator&lt;wcha r_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, wchar_t, tm const*, char,</pre>

	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 9-334

**Table 9-334 typeinfo for `time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>time_put_byname&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code>

### 9.1.116.2 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 9-335, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-335 libstdcxx - Class `time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >` Function Interfaces**

<code>time_put_byname&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::~time_put_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_put_byname&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::~time_put_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_put_byname&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::~time_put_byname()(GLIBCXX_3.4) [ISOCXX]</code>

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 9-336, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-336 libstdcxx - Class `time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >` Data Interfaces**

<code>typeinfo for time_put_byname&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeinfo name for time_put_byname&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for time_put_byname&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.117 Class `time_get<char, istreambuf_iterator<char, char_traits<char>>>`

#### 9.1.117.1 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 9-337

**Table 9-337 Primary vtable for `time_get<char, istreambuf_iterator<char, char_traits<char>>>`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeinfo for time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code>
<code>vfunc[0]:</code>	<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::~~time_get()</code>
<code>vfunc[1]:</code>	<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::~~time_get()</code>
<code>vfunc[2]:</code>	<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::do_date_order() const</code>
<code>vfunc[3]:</code>	<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::do_get_time(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;, tm*) const</code>
<code>vfunc[4]:</code>	<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::do_get_date(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;, tm*) const</code>
<code>vfunc[5]:</code>	<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::do_get_weekday(istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, ios_base&amp;</code>



	<code>_Ios_Iostate&amp;, tm*) const</code>
<code>vfunc[6]:</code>	<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_get_monthname(istreambuf_it erator&lt;char, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, tm*) const</code>
<code>vfunc[7]:</code>	<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_get_year(istreambuf_iterator&lt;c har, char_traits&lt;char&gt; &gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, tm*) const</code>

### 9.1.117.2 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 9-338, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-338 libstdcxx - Class `time_get<char, istreambuf_iterator<char, char_traits<char> > >` Function Interfaces**

<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::date_order() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_get_date(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;, tm*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_get_time(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;, tm*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_get_year(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;, tm*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::get_weekday(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;, tm*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_date_order() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;</code>

<code>&gt;::get_monthname(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;, tm*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_get_weekday(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;, tm*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::do_get_monthname(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;, tm*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::M_extract_via_format(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;, tm*, char const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::get_date(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;, tm*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::time(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;, tm*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::get_year(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;, tm*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::~time_get()(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::~time_get()(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::~time_get()(GLIBCXX_3.4) [ISOCXX]</code>

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 9-339, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-339 libstdcxx - Class `time_get<char, istreambuf_iterator<char, char_traits<char> > >` Data Interfaces**

<code>guard variable for <code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::id(GLIBCXX_3.4) [CXXABI]</code></code>
<code><code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::id(GLIBCXX_3.4) [ISOCXX]</code></code>
<code><code>typeid for <code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;</code>(GLIBCXX_3.4) [CXXABI]</code></code>

typeinfo name for <code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtbl for <code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.118 Class `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

#### 9.1.118.1 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 9-340

**Table 9-340 Primary vtable for `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code>
vfunc[0]:	<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::~time_get()</code>
vfunc[1]:	<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::~time_get()</code>
vfunc[2]:	<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::do_date_order() const</code>
vfunc[3]:	<code>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</code>
vfunc[4]:	<code>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;,</code>

	<code>_Ios_Iostate&amp;, tm*) const</code>
<code>vfunc[5]:</code>	<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::do_get_weekday(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;, ios_base&amp;, _Ios_Iostate&amp;, tm*) const</code>
<code>vfunc[6]:</code>	<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::do_get_monthname(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;, ios_base&amp;, _Ios_Iostate&amp;, tm*) const</code>
<code>vfunc[7]:</code>	<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::do_get_year(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;, ios_base&amp;, _Ios_Iostate&amp;, tm*) const</code>

### 9.1.118.2 Interfaces for Class `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

An LSB conforming implementation shall provide the generic methods for Class `std::time_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t>>>` specified in Table 9-341, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-341 libstdcxx - Class `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>` Function Interfaces**

<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::date_order() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>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(GLIBCXX_3.4) [ISOCXX]</code>
<code>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(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::do_get_year(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;,</code>

<code>_Ios_Iostate&amp;, tm*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::get_weekday(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(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::do_date_order() const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::get_monthname(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(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::do_get_weekday(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(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::do_get_monthname(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(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::M_extract_via_format(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*, wchar_t const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::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(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::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(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::get_year(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(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::~time_get()(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::~time_get()(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::~time_get()(GLIBCXX_3.4) [ISOCXX]</code>

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 9-342, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-342 libstdcxx - Class `time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>` Data Interfaces**

guard variable for <code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::id(GLIBCXX_3.4)</code> [CXXABI]
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::id(GLIBCXX_3.4)</code> [ISOCXX]
typeid for <code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]

**9.1.119 Class `time_put<char, ostreambuf_iterator<char, char_traits<char>>>`****9.1.119.1 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 9-343, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-343 libstdcxx - Class `time_put<char, ostreambuf_iterator<char, char_traits<char>>>` Function Interfaces**

<code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, ios_base&amp;, char, tm const*, char const*, char const*)</code> const(GLIBCXX_3.4) [ISOCXX]
<code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, ios_base&amp;, char, tm const*, char, char)</code> const(GLIBCXX_3.4) [ISOCXX]
<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)</code> const(GLIBCXX_3.4) [ISOCXX]
<code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::~~time_put()</code> (GLIBCXX_3.4) [ISOCXX]
<code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::~~time_put()</code> (GLIBCXX_3.4) [ISOCXX]
<code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::~~time_put()</code> (GLIBCXX_3.4) [ISOCXX]

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 9-344, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-344 libstdcxx - Class `time_put<char, ostreambuf_iterator<char, char_traits<char>>>` Data Interfaces**

guard variable for <code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::id(GLIBCXX_3.4)</code> [CXXABI]
<code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::id(GLIBCXX_3.4)</code> [ISOCXX]
typeid for <code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.120 Class `time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

#### 9.1.120.1 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 9-345, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-345 libstdcxx - Class `time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>` Function Interfaces**

<code>time_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;, ios_base&amp;, wchar_t, tm const*, wchar_t const*, wchar_t const*) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>time_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;, ios_base&amp;, wchar_t, tm const*, char, char) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>time_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;, ios_base&amp;, wchar_t, tm const*, char, char) const</code> (GLIBCXX_3.4) [ISOCXX]
<code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::~time_put()</code> (GLIBCXX_3.4) [ISOCXX]
<code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::~time_put()</code> (GLIBCXX_3.4) [ISOCXX]
<code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::~time_put()</code> (GLIBCXX_3.4) [ISOCXX]

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 9-346, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-346 libstdcxx - Class `time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t>>>` Data Interfaces**

guard variable for <code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::id(GLIBCXX_3.4)</code> [CXXABI]
<code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;::id(GLIBCXX_3.4)</code> [ISOCXX]
typeid for <code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]

**9.1.121 Class `money_punct<char, false>`****9.1.121.1 Class `data` for `money_punct<char, false>`**

The virtual table for the `std::money_punct<char, false>` class is described by Table 9-347

**Table 9-347 Primary vtable for `money_punct<char, false>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>money_punct&lt;char, false&gt;</code>
<code>vfunc[0]:</code>	<code>money_punct&lt;char, false&gt;::~~money_punct()</code>
<code>vfunc[1]:</code>	<code>money_punct&lt;char, false&gt;::~~money_punct()</code>
<code>vfunc[2]:</code>	<code>money_punct&lt;char, false&gt;::do_decimal_point() const</code>
<code>vfunc[3]:</code>	<code>money_punct&lt;char, false&gt;::do_thousands_sep() const</code>
<code>vfunc[4]:</code>	<code>money_punct&lt;char, false&gt;::do_grouping() const</code>
<code>vfunc[5]:</code>	<code>money_punct&lt;char, false&gt;::do_curr_symbol() const</code>
<code>vfunc[6]:</code>	<code>money_punct&lt;char, false&gt;::do_positive_sign() const</code>
<code>vfunc[7]:</code>	<code>money_punct&lt;char, false&gt;::do_negative_sign() const</code>
<code>vfunc[8]:</code>	<code>money_punct&lt;char, false&gt;::do_frac_digits() const</code>
<code>vfunc[9]:</code>	<code>money_punct&lt;char,</code>



	false>::do_pos_format() const
vfunc[10]:	money_punct<char, false>::do_neg_format() const

### 9.1.121.2 Interfaces for Class money\_punct<char, false>

An LSB conforming implementation shall provide the generic methods for Class `std::money_punct<char, false>` specified in Table 9-348, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-348 libstdcxx - Class money\_punct<char, false> Function Interfaces**

money_punct<char, false>::neg_format() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::pos_format() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::curr_symbol() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::do_grouping() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::frac_digits() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::decimal_point() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::do_neg_format() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::do_pos_format() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::negative_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::positive_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::thousands_sep() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::do_curr_symbol() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::do_frac_digits() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::do_decimal_point() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::do_negative_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::do_positive_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::do_thousands_sep() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::grouping() const(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::_M_initialize_money_punct(__locale_struct*, char const*)(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::~money_punct()(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::~money_punct()(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::~money_punct()(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::money_punct<char, false>` specified in Table 9-349, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-349 libstdcxx - Class money\_punct<char, false> Data Interfaces**

guard variable for money_punct<char, false>::id(GLIBCXX_3.4) [CXXABI]
money_punct<char, false>::id(GLIBCXX_3.4) [ISOCXX]
money_punct<char, false>::intl(GLIBCXX_3.4) [ISOCXX]
typeinfo for money_punct<char, false>(GLIBCXX_3.4) [CXXABI]
typeinfo name for money_punct<char, false>(GLIBCXX_3.4) [CXXABI]
vtable for money_punct<char, false>(GLIBCXX_3.4) [CXXABI]

**9.1.122 Class money\_punct<char, true>****9.1.122.1 Class data for money\_punct<char, true>**

The virtual table for the std::money\_punct<char, true> class is described by Table 9-350

**Table 9-350 Primary vtable for money\_punct<char, true>**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for money_punct<char, true>
vfunc[0]:	money_punct<char, true>::~~money_punct()
vfunc[1]:	money_punct<char, true>::~~money_punct()
vfunc[2]:	money_punct<char, true>::do_decimal_point() const
vfunc[3]:	money_punct<char, true>::do_thousands_sep() const
vfunc[4]:	money_punct<char, true>::do_grouping() const
vfunc[5]:	money_punct<char, true>::do_curr_symbol() const
vfunc[6]:	money_punct<char, true>::do_positive_sign() const
vfunc[7]:	money_punct<char, true>::do_negative_sign() const
vfunc[8]:	money_punct<char, true>::do_frac_digits() const
vfunc[9]:	money_punct<char, true>::do_pos_format() const
vfunc[10]:	money_punct<char, true>::do_neg_format() const

### 9.1.122.2 Interfaces for Class `money_punct<char, true>`

An LSB conforming implementation shall provide the generic methods for Class `std::money_punct<char, true>` specified in Table 9-351, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-351 libstdcxx - Class `money_punct<char, true>` Function Interfaces**

<code>money_punct&lt;char, true&gt;::neg_format()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::pos_format()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::curr_symbol()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::do_grouping()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::frac_digits()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::decimal_point()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::do_neg_format()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::do_pos_format()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::negative_sign()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::positive_sign()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::thousands_sep()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::do_curr_symbol()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::do_frac_digits()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::do_decimal_point()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::do_negative_sign()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::do_positive_sign()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::do_thousands_sep()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::grouping()</code> const(GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::M_initialize_money_punct(__locale_struct*, char const*)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_punct&lt;char, true&gt;::~money_punct()</code> (GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::~money_punct()</code> (GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::~money_punct()</code> (GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::money_punct<char, true>` specified in Table 9-352, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-352 libstdcxx - Class `money_punct<char, true>` Data Interfaces**

guard variable for <code>money_punct&lt;char, true&gt;::id</code> (GLIBCXX_3.4) [CXXABI]
<code>money_punct&lt;char, true&gt;::id</code> (GLIBCXX_3.4) [ISOCXX]
<code>money_punct&lt;char, true&gt;::intl</code> (GLIBCXX_3.4) [ISOCXX]
<code>typeid</code> for <code>money_punct&lt;char, true&gt;</code> (GLIBCXX_3.4) [CXXABI]

typeid name for <code>money_punct&lt;char, true&gt;(GLIBCXX_3.4)</code> [CXXABI]
vtable for <code>money_punct&lt;char, true&gt;(GLIBCXX_3.4)</code> [CXXABI]

### 9.1.123 Class `money_punct<wchar_t, false>`

#### 9.1.123.1 Class data for `money_punct<wchar_t, false>`

The virtual table for the `std::money_punct<wchar_t, false>` class is described by Table 9-353

**Table 9-353 Primary vtable for `money_punct<wchar_t, false>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>money_punct&lt;wchar_t, false&gt;</code>
<code>vfunc[0]:</code>	<code>money_punct&lt;wchar_t, false&gt;::~~money_punct()</code>
<code>vfunc[1]:</code>	<code>money_punct&lt;wchar_t, false&gt;::~~money_punct()</code>
<code>vfunc[2]:</code>	<code>money_punct&lt;wchar_t, false&gt;::~do_decimal_point() const</code>
<code>vfunc[3]:</code>	<code>money_punct&lt;wchar_t, false&gt;::~do_thousands_sep() const</code>
<code>vfunc[4]:</code>	<code>money_punct&lt;wchar_t, false&gt;::~do_grouping() const</code>
<code>vfunc[5]:</code>	<code>money_punct&lt;wchar_t, false&gt;::~do_curr_symbol() const</code>
<code>vfunc[6]:</code>	<code>money_punct&lt;wchar_t, false&gt;::~do_positive_sign() const</code>
<code>vfunc[7]:</code>	<code>money_punct&lt;wchar_t, false&gt;::~do_negative_sign() const</code>
<code>vfunc[8]:</code>	<code>money_punct&lt;wchar_t, false&gt;::~do_frac_digits() const</code>
<code>vfunc[9]:</code>	<code>money_punct&lt;wchar_t, false&gt;::~do_pos_format() const</code>
<code>vfunc[10]:</code>	<code>money_punct&lt;wchar_t, false&gt;::~do_neg_format() const</code>

#### 9.1.123.2 Interfaces for Class `money_punct<wchar_t, false>`

An LSB conforming implementation shall provide the generic methods for Class `std::money_punct<wchar_t, false>` specified in Table 9-354, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-354 libstdc++ - Class `money_punct<wchar_t, false>` Function Interfaces**

<code>money_punct&lt;wchar_t, false&gt;::~neg_format() const(GLIBCXX_3.4)</code> [ISOCXX]
---

money_punct<wchar_t, false>::pos_format() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::curr_symbol() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::do_grouping() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::frac_digits() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::decimal_point() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::do_neg_format() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::do_pos_format() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::negative_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::positive_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::thousands_sep() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::do_curr_symbol() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::do_frac_digits() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::do_decimal_point() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::do_negative_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::do_positive_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::do_thousands_sep() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::grouping() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::_M_initialize_money_punct(__locale_struct*, char const*)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::~money_punct()(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::~money_punct()(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, false>::~money_punct()(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::money_punct<wchar_t, false>` specified in Table 9-355, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-355 libstdc++ - Class `money_punct<wchar_t, false>` Data Interfaces**

guard variable for <code>money_punct&lt;wchar_t, false&gt;::id(GLIBCXX_3.4)</code> [CXXABI]
<code>money_punct&lt;wchar_t, false&gt;::id(GLIBCXX_3.4)</code> [ISOCXX]
<code>money_punct&lt;wchar_t, false&gt;::intl(GLIBCXX_3.4)</code> [ISOCXX]
<code>typeid</code> for <code>money_punct&lt;wchar_t, false&gt;</code> (GLIBCXX_3.4) [CXXABI]

typeid name for <code>money_punct&lt;wchar_t, false&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>money_punct&lt;wchar_t, false&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.124 Class `money_punct<wchar_t, true>`

#### 9.1.124.1 Class data for `money_punct<wchar_t, true>`

The virtual table for the `std::money_punct<wchar_t, true>` class is described by Table 9-356

**Table 9-356 Primary vtable for `money_punct<wchar_t, true>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for <code>money_punct&lt;wchar_t, true&gt;</code>
<code>vfunc[0]:</code>	<code>money_punct&lt;wchar_t, true&gt;::~~money_punct()</code>
<code>vfunc[1]:</code>	<code>money_punct&lt;wchar_t, true&gt;::~~money_punct()</code>
<code>vfunc[2]:</code>	<code>money_punct&lt;wchar_t, true&gt;::do_decimal_point() const</code>
<code>vfunc[3]:</code>	<code>money_punct&lt;wchar_t, true&gt;::do_thousands_sep() const</code>
<code>vfunc[4]:</code>	<code>money_punct&lt;wchar_t, true&gt;::do_grouping() const</code>
<code>vfunc[5]:</code>	<code>money_punct&lt;wchar_t, true&gt;::do_curr_symbol() const</code>
<code>vfunc[6]:</code>	<code>money_punct&lt;wchar_t, true&gt;::do_positive_sign() const</code>
<code>vfunc[7]:</code>	<code>money_punct&lt;wchar_t, true&gt;::do_negative_sign() const</code>
<code>vfunc[8]:</code>	<code>money_punct&lt;wchar_t, true&gt;::do_frac_digits() const</code>
<code>vfunc[9]:</code>	<code>money_punct&lt;wchar_t, true&gt;::do_pos_format() const</code>
<code>vfunc[10]:</code>	<code>money_punct&lt;wchar_t, true&gt;::do_neg_format() const</code>

#### 9.1.124.2 Interfaces for Class `money_punct<wchar_t, true>`

An LSB conforming implementation shall provide the generic methods for Class `std::money_punct<wchar_t, true>` specified in Table 9-357, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-357 libstdc++ - Class `money_punct<wchar_t, true>` Function Interfaces**

<code>money_punct&lt;wchar_t, true&gt;::neg_format() const</code> (GLIBCXX_3.4) [ISOCXX]
--

money_punct<wchar_t, true>::pos_format() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::curr_symbol() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::do_grouping() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::frac_digits() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::decimal_point() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::do_neg_format() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::do_pos_format() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::negative_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::positive_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::thousands_sep() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::do_curr_symbol() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::do_frac_digits() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::do_decimal_point() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::do_negative_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::do_positive_sign() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::do_thousands_sep() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::grouping() const(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::_M_initialize_money_punct(__locale_struct*, char const*)(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::~money_punct()(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::~money_punct()(GLIBCXX_3.4) [ISOCXX]
money_punct<wchar_t, true>::~money_punct()(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::money_punct<wchar_t, true>` specified in Table 9-358, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-358 libstdc++ - Class `money_punct<wchar_t, true>` Data Interfaces**

guard variable for <code>money_punct&lt;wchar_t, true&gt;::id(GLIBCXX_3.4)</code> [CXXABI]
<code>money_punct&lt;wchar_t, true&gt;::id(GLIBCXX_3.4)</code> [ISOCXX]
<code>money_punct&lt;wchar_t, true&gt;::intl(GLIBCXX_3.4)</code> [ISOCXX]
typeinfo for <code>money_punct&lt;wchar_t, true&gt;</code> (GLIBCXX_3.4) [CXXABI]

typeid name for <code>money_punct&lt;wchar_t, true&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>money_punct&lt;wchar_t, true&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.125 Class `money_punct_byname<char, false>`

#### 9.1.125.1 Class data for `money_punct_byname<char, false>`

The virtual table for the `std::money_punct_byname<char, false>` class is described by Table 9-359

**Table 9-359 Primary vtable for `money_punct_byname<char, false>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid name for <code>money_punct_byname&lt;char, false&gt;</code>
<code>vfunc[0]:</code>	<code>money_punct_byname&lt;char, false&gt;::~~money_punct_byname()</code>
<code>vfunc[1]:</code>	<code>money_punct_byname&lt;char, false&gt;::~~money_punct_byname()</code>
<code>vfunc[2]:</code>	<code>money_punct&lt;char, false&gt;::~do_decimal_point() const</code>
<code>vfunc[3]:</code>	<code>money_punct&lt;char, false&gt;::~do_thousands_sep() const</code>
<code>vfunc[4]:</code>	<code>money_punct&lt;char, false&gt;::~do_grouping() const</code>
<code>vfunc[5]:</code>	<code>money_punct&lt;char, false&gt;::~do_curr_symbol() const</code>
<code>vfunc[6]:</code>	<code>money_punct&lt;char, false&gt;::~do_positive_sign() const</code>
<code>vfunc[7]:</code>	<code>money_punct&lt;char, false&gt;::~do_negative_sign() const</code>
<code>vfunc[8]:</code>	<code>money_punct&lt;char, false&gt;::~do_frac_digits() const</code>
<code>vfunc[9]:</code>	<code>money_punct&lt;char, false&gt;::~do_pos_format() const</code>
<code>vfunc[10]:</code>	<code>money_punct&lt;char, false&gt;::~do_neg_format() const</code>

The Run Time Type Information for the `std::money_punct_byname<char, false>` class is described by Table 9-360

**Table 9-360 typeid name for `money_punct_byname<char, false>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
-------------	--



Name	typeinfo name for money_punct_byname<char, false>
------	--

### 9.1.125.2 Interfaces for Class `money_punct_byname<char, false>`

An LSB conforming implementation shall provide the generic methods for Class `std::money_punct_byname<char, false>` specified in Table 9-361, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-361 libstdcxx - Class `money_punct_byname<char, false>` Function Interfaces**

<code>money_punct_byname&lt;char, false&gt;::~~money_punct_byname()(GLIBCXX_3.4)</code> [ISOCXX]
<code>money_punct_byname&lt;char, false&gt;::~~money_punct_byname()(GLIBCXX_3.4)</code> [ISOCXX]
<code>money_punct_byname&lt;char, false&gt;::~~money_punct_byname()(GLIBCXX_3.4)</code> [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::money_punct_byname<char, false>` specified in Table 9-362, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-362 libstdcxx - Class `money_punct_byname<char, false>` Data Interfaces**

<code>money_punct_byname&lt;char, false&gt;::intl(GLIBCXX_3.4)</code> [ISOCXX]
typeinfo for <code>money_punct_byname&lt;char, false&gt;(GLIBCXX_3.4)</code> [CXXABI]
typeinfo name for <code>money_punct_byname&lt;char, false&gt;(GLIBCXX_3.4)</code> [CXXABI]
vtable for <code>money_punct_byname&lt;char, false&gt;(GLIBCXX_3.4)</code> [CXXABI]

### 9.1.126 Class `money_punct_byname<char, true>`

#### 9.1.126.1 Class data for `money_punct_byname<char, true>`

The virtual table for the `std::money_punct_byname<char, true>` class is described by Table 9-363

**Table 9-363 Primary vtable for `money_punct_byname<char, true>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>money_punct_byname&lt;char, true&gt;</code>
<code>vfunc[0]:</code>	<code>money_punct_byname&lt;char, true&gt;::~~money_punct_byname()</code>
<code>vfunc[1]:</code>	<code>money_punct_byname&lt;char, true&gt;::~~money_punct_byname()</code>

vfunc[2]:	money_punct<char, true>::do_decimal_point() const
vfunc[3]:	money_punct<char, true>::do_thousands_sep() const
vfunc[4]:	money_punct<char, true>::do_grouping() const
vfunc[5]:	money_punct<char, true>::do_curr_symbol() const
vfunc[6]:	money_punct<char, true>::do_positive_sign() const
vfunc[7]:	money_punct<char, true>::do_negative_sign() const
vfunc[8]:	money_punct<char, true>::do_frac_digits() const
vfunc[9]:	money_punct<char, true>::do_pos_format() const
vfunc[10]:	money_punct<char, true>::do_neg_format() const

The Run Time Type Information for the `std::money_punct_byname<char, true>` class is described by Table 9-364

**Table 9-364 typeinfo for `money_punct_byname<char, true>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>money_punct_byname&lt;char, true&gt;</code>

### 9.1.126.2 Interfaces for Class `money_punct_byname<char, true>`

An LSB conforming implementation shall provide the generic methods for Class `std::money_punct_byname<char, true>` specified in Table 9-365, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-365 libstdc++ - Class `money_punct_byname<char, true>` Function Interfaces**

<code>money_punct_byname&lt;char, true&gt;::~~money_punct_byname()(GLIBCXX_3.4)</code> [ISOCXX]
<code>money_punct_byname&lt;char, true&gt;::~~money_punct_byname()(GLIBCXX_3.4)</code> [ISOCXX]
<code>money_punct_byname&lt;char, true&gt;::~~money_punct_byname()(GLIBCXX_3.4)</code> [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::money_punct_byname<char, true>` specified in Table 9-366, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-366 libstdcxx - Class money\_punct\_byname<char, true> Data Interfaces**

money_punct_byname<char, true>::intl(GLIBCXX_3.4) [ISOCXX]
typeid for money_punct_byname<char, true>(GLIBCXX_3.4) [CXXABI]
typeid name for money_punct_byname<char, true>(GLIBCXX_3.4) [CXXABI]
vtable for money_punct_byname<char, true>(GLIBCXX_3.4) [CXXABI]

**9.1.127 Class money\_punct\_byname<wchar\_t, false>****9.1.127.1 Class data for money\_punct\_byname<wchar\_t, false>**

The virtual table for the `std::money_punct_byname<wchar_t, false>` class is described by Table 9-367

**Table 9-367 Primary vtable for money\_punct\_byname<wchar\_t, false>**

Base Offset	0
Virtual Base Offset	0
RTTI	typeid for money_punct_byname<wchar_t, false>
vfunc[0]:	money_punct_byname<wchar_t, false>::~~money_punct_byname()
vfunc[1]:	money_punct_byname<wchar_t, false>::~~money_punct_byname()
vfunc[2]:	money_punct<wchar_t, false>::~do_decimal_point() const
vfunc[3]:	money_punct<wchar_t, false>::~do_thousands_sep() const
vfunc[4]:	money_punct<wchar_t, false>::~do_grouping() const
vfunc[5]:	money_punct<wchar_t, false>::~do_curr_symbol() const
vfunc[6]:	money_punct<wchar_t, false>::~do_positive_sign() const
vfunc[7]:	money_punct<wchar_t, false>::~do_negative_sign() const
vfunc[8]:	money_punct<wchar_t, false>::~do_frac_digits() const
vfunc[9]:	money_punct<wchar_t, false>::~do_pos_format() const
vfunc[10]:	money_punct<wchar_t, false>::~do_neg_format() const

The Run Time Type Information for the `std::moneypunct_byname<wchar_t, false>` class is described by Table 9-368

**Table 9-368 typeid for `moneypunct_byname<wchar_t, false>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeid name for <code>moneypunct_byname&lt;wchar_t, false&gt;</code>

### 9.1.127.2 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 9-369, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-369 libstdcxx - Class `moneypunct_byname<wchar_t, false>` Function Interfaces**

<code>moneypunct_byname&lt;wchar_t, false&gt;::~moneypunct_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>moneypunct_byname&lt;wchar_t, false&gt;::~moneypunct_byname()(GLIBCXX_3.4) [ISOCXX]</code>
<code>moneypunct_byname&lt;wchar_t, false&gt;::~moneypunct_byname()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::moneypunct_byname<wchar_t, false>` specified in Table 9-370, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-370 libstdcxx - Class `moneypunct_byname<wchar_t, false>` Data Interfaces**

<code>moneypunct_byname&lt;wchar_t, false&gt;::intl(GLIBCXX_3.4) [ISOCXX]</code>
typeid for <code>moneypunct_byname&lt;wchar_t, false&gt;(GLIBCXX_3.4) [CXXABI]</code>
typeid name for <code>moneypunct_byname&lt;wchar_t, false&gt;(GLIBCXX_3.4) [CXXABI]</code>
vtable for <code>moneypunct_byname&lt;wchar_t, false&gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.128 Class `moneypunct_byname<wchar_t, true>`

#### 9.1.128.1 Class data for `moneypunct_byname<wchar_t, true>`

The virtual table for the `std::moneypunct_byname<wchar_t, true>` class is described by Table 9-371

**Table 9-371 Primary vtable for `moneypunct_byname<wchar_t, true>`**

Base Offset	0
Virtual Base Offset	0

RTTI	typeid for moneypunct_byname<wchar_t, true>
vfunc[0]:	moneypunct_byname<wchar_t, true>::~~moneypunct_byname()
vfunc[1]:	moneypunct_byname<wchar_t, true>::~~moneypunct_byname()
vfunc[2]:	moneypunct<wchar_t, true>::do_decimal_point() const
vfunc[3]:	moneypunct<wchar_t, true>::do_thousands_sep() const
vfunc[4]:	moneypunct<wchar_t, true>::do_grouping() const
vfunc[5]:	moneypunct<wchar_t, true>::do_curr_symbol() const
vfunc[6]:	moneypunct<wchar_t, true>::do_positive_sign() const
vfunc[7]:	moneypunct<wchar_t, true>::do_negative_sign() const
vfunc[8]:	moneypunct<wchar_t, true>::do_frac_digits() const
vfunc[9]:	moneypunct<wchar_t, true>::do_pos_format() const
vfunc[10]:	moneypunct<wchar_t, true>::do_neg_format() const

The Run Time Type Information for the `std::moneypunct_byname<wchar_t, true>` class is described by Table 9-372

**Table 9-372 typeid for moneypunct\_byname<wchar\_t, true>**

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeid name for moneypunct_byname<wchar_t, true>

### 9.1.128.2 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 9-373, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-373 libstdcxx - Class `moneypunct_byname<wchar_t, true>` Function Interfaces**

<code>moneypunct_byname&lt;wchar_t, true&gt;::~~moneypunct_byname()</code> (GLIBCXX_3.4) [ISOCXX]
---

money_punct_byname<wchar_t, true>::~~money_punct_byname()(GLIBCXX_3.4) [ISOCXX]
--

money_punct_byname<wchar_t, true>::~~money_punct_byname()(GLIBCXX_3.4) [ISOCXX]
--

An LSB conforming implementation shall provide the generic data interfaces for Class `std::money_punct_byname<wchar_t, true>` specified in Table 9-374, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-374 libstdc++ - Class `money_punct_byname<wchar_t, true>` Data Interfaces**

money_punct_byname<wchar_t, true>::intl(GLIBCXX_3.4) [ISOCXX]
---

typeid for money_punct_byname<wchar_t, true>(GLIBCXX_3.4) [CXXABI]
--

typeid name for money_punct_byname<wchar_t, true>(GLIBCXX_3.4) [CXXABI]
---

vtable for money_punct_byname<wchar_t, true>(GLIBCXX_3.4) [CXXABI]
--

### 9.1.129 Class `money_base`

#### 9.1.129.1 Class data for `money_base`

The Run Time Type Information for the `std::money_base` class is described by Table 9-375

**Table 9-375 typeid for `money_base`**

Base Vtable	vtable for <code>__cxxabiv1::__class_type_info</code>
Name	typeid name for <code>money_base</code>

#### 9.1.129.2 Interfaces for Class `money_base`

An LSB conforming implementation shall provide the generic methods for Class `std::money_base` specified in Table 9-376, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-376 libstdc++ - Class `money_base` Function Interfaces**

<code>money_base::_S_construct_pattern(char, char, char)(GLIBCXX_3.4) [ISOCXX]</code>
---

An LSB conforming implementation shall provide the generic data interfaces for Class `std::money_base` specified in Table 9-377, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-377 libstdc++ - Class `money_base` Data Interfaces**

<code>money_base::_S_default_pattern(GLIBCXX_3.4) [ISOCXX]</code>
---

<code>money_base::_S_atoms(GLIBCXX_3.4) [ISOCXX]</code>
---

typeid for <code>money_base(GLIBCXX_3.4) [CXXABI]</code>
--

typeid name for <code>money_base(GLIBCXX_3.4) [CXXABI]</code>
---

### 9.1.130 Class `money_get<char, istreambuf_iterator<char, char_traits<char>>>`

#### 9.1.130.1 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 9-378

**Table 9-378 Primary vtable for `money_get<char, istreambuf_iterator<char, char_traits<char>>>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code>
vfunc[0]:	<code>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::~~money_get()</code>
vfunc[1]:	<code>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::~~money_get()</code>
vfunc[2]:	<code>money_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;, bool, ios_base&amp;, _Ios_Iostate&amp;, long double&amp;) const</code>
vfunc[3]:	<code>money_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;, bool, ios_base&amp;, _Ios_Iostate&amp;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&amp;) const</code>

The Run Time Type Information for the `std::money_get<char, std::istreambuf_iterator<char, std::char_traits<char>>>` class is described by Table 9-379

**Table 9-379 typeinfo for `money_get<char, istreambuf_iterator<char, char_traits<char>>>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
-------------	--

Name	typeinfo name for <code>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code>
------	--

### 9.1.130.2 Interfaces for Class `money_get<char, istreambuf_iterator<char, char_traits<char>>>`

An LSB conforming implementation shall provide the generic methods for Class `std::money_get<char, std::istreambuf_iterator<char, std::char_traits<char>>>` specified in Table 9-380, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-380 libstdcxx - Class `money_get<char, istreambuf_iterator<char, char_traits<char>>>` Function Interfaces**

<pre>istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;::_M_extract&lt;false&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;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;::_M_extract&lt;true&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;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;::get(istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, bool, 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]</pre>
<pre>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;::get(istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, bool, ios_base&amp;, _Ios_Iostate&amp;, long double&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&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;, bool, 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]</pre>
<pre>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&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;, bool, ios_base&amp;, _Ios_Iostate&amp;, long double&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;::~money_get()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt; &gt;::~money_get()(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; &gt;</pre>



>::~money_get()(GLIBCXX_3.4) [ISOCXX]
---------------------------------------

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 9-381, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-381 libstdcxx - Class `money_get<char, istreambuf_iterator<char, char_traits<char>>>` Data Interfaces**

guard variable for <code>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> :>::id(GLIBCXX_3.4) [CXXABI]
<code>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> :>::id(GLIBCXX_3.4) [ISOCXX]
typeinfo for <code>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> :>(GLIBCXX_3.4) [CXXABI]
typeinfo name for <code>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> :>(GLIBCXX_3.4) [CXXABI]
vtable for <code>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> :>(GLIBCXX_3.4) [CXXABI]

### 9.1.131 Class `money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`

#### 9.1.131.1 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 9-382

**Table 9-382 Primary vtable for `money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>`**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for <code>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code>
vfunc[0]:	<code>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code> :>::~money_get()
vfunc[1]:	<code>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code> :>::~money_get()
vfunc[2]:	<code>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;</code>

	<pre>&gt;::do_get(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, bool, ios_base&amp;, _Ios_Iostate&amp;, long double&amp;) const</pre>
vfunc[3]:	<pre>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;::do_get(istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, bool, ios_base&amp;, _Ios_Iostate&amp;, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;, allocator&lt;wchar_t&gt; &gt;&amp;) const</pre>

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 9-383

**Table 9-383 typeinfo for `money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code>

### 9.1.131.2 Interfaces for Class `money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >`

An LSB conforming implementation shall provide the generic methods for Class `std::money_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > >` specified in Table 9-384, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-384 libstdcxx - Class `money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >` Function Interfaces**

<pre>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;::_M_extract&lt;false&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;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;::_M_extract&lt;true&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;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&amp;)</pre>

const(GLIBCXX_3.4) [ISOCXX]
money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>::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(GLIBCXX_3.4) [ISOCXX]
money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>::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(GLIBCXX_3.4) [ISOCXX]
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(GLIBCXX_3.4) [ISOCXX]
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(GLIBCXX_3.4) [ISOCXX]
money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>::~~money_get()(GLIBCXX_3.4) [ISOCXX]
money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>::~~money_get()(GLIBCXX_3.4) [ISOCXX]
money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>::~~money_get()(GLIBCXX_3.4) [ISOCXX]

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 9-385, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-385 libstdcxx - Class `money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>` Data Interfaces**

guard variable for <code>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::id(GLIBCXX_3.4)</code> [CXXABI]
<code>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::id(GLIBCXX_3.4)</code> [ISOCXX]
typeid for <code>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;</code> (GLIBCXX_3.4) [CXXABI]

### 9.1.132 Class `money_put<char, ostreambuf_iterator<char, char_traits<char>>>`

#### 9.1.132.1 Class data for `money_put<char, ostreambuf_iterator<char, char_traits<char>>>`

The virtual table for the `std::money_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>>` class is described by Table 9-386

**Table 9-386 Primary vtable for `money_put<char, ostreambuf_iterator<char, char_traits<char>>>`**

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeinfo</code> for <code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code>
<code>vfunc[0]:</code>	<code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::~~money_put()</code>
<code>vfunc[1]:</code>	<code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;::~~money_put()</code>
<code>vfunc[2]:</code>	<code>money_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;, bool, ios_base&amp;, char, long double) const</code>
<code>vfunc[3]:</code>	<code>money_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;, bool, ios_base&amp;, char, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;) const</code>

The Run Time Type Information for the `std::money_put<char, std::ostreambuf_iterator<char, std::char_traits<char>>>` class is described by Table 9-387

**Table 9-387 `typeinfo` for `money_put<char, ostreambuf_iterator<char, char_traits<char>>>`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	<code>typeinfo</code> name for <code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code>

### 9.1.132.2 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 9-388, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-388 libstdcxx - Class `money_put<char, ostreambuf_iterator<char, char_traits<char>>>` Function Interfaces**

<code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, bool, ios_base&amp;, char,</code> <code>basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, bool, ios_base&amp;, char,</code> <code>long double) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::do_put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, bool, ios_base&amp;,</code> <code>char, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt; const&amp;)</code> <code>const(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::do_put(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;, bool, ios_base&amp;,</code> <code>char, long double) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; money_put&lt;char,</code> <code>ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;</code> <code>&gt;::_M_insert&lt;false&gt;(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; ,</code> <code>ios_base&amp;, char, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;</code> <code>const&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; money_put&lt;char,</code> <code>ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;</code> <code>&gt;::_M_insert&lt;true&gt;(ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; ,</code> <code>ios_base&amp;, char, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt;&gt;</code> <code>const&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::~~money_put()(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::~~money_put()(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;</code> <code>&gt;::~~money_put()(GLIBCXX_3.4) [ISOCXX]</code>

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 9-389, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-389 libstdcxx - Class `money_put<char, ostreambuf_iterator<char, char_traits<char>>>` Data Interfaces**

<code>guard variable for money_put&lt;char, ostreambuf_iterator&lt;char,</code>
---

<code>char_traits&lt;char&gt; &gt; &gt;::id(GLIBCXX_3.4) [CXXABI]</code>
<code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::id(GLIBCXX_3.4) [ISOCXX]</code>
<code>typeid for money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.133 Class `money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >`

#### 9.1.133.1 Class data for `money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >`

The virtual table for the `std::money_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > >` class is described by Table 9-390

Table 9-390 Primary vtable for `money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >`

Base Offset	0
Virtual Base Offset	0
RTTI	<code>typeid for money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code>
<code>vfunc[0]:</code>	<code>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::~~money_put()</code>
<code>vfunc[1]:</code>	<code>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::~~money_put()</code>
<code>vfunc[2]:</code>	<code>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</code>
<code>vfunc[3]:</code>	<code>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,</code>

	basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&) const
--	--

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 9-391

**Table 9-391 typeinfo for `money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >`**

Base Vtable	vtable for <code>__cxxabiv1::__si_class_type_info</code>
Name	typeinfo name for <code>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code>

### 9.1.133.2 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 9-392, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-392 libstdc++ - Class `money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >` Function Interfaces**

<code>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) [ISOCXX]</code>
<code>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, long double) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>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(GLIBCXX_3.4) [ISOCXX]</code>
<code>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(GLIBCXX_3.4) [ISOCXX]</code>
<code>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::_M_insert&lt;false&gt;(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, 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) [ISOCXX]</code>
<code>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::_M_insert&lt;true&gt;(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, wchar_t, basic_string&lt;wchar_t, char_traits&lt;wchar_t&gt;,</code>

allocator<wchar_t> > const&) const(GLIBCXX_3.4) [ISOCXX]
money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::~money_put()(GLIBCXX_3.4) [ISOCXX]
money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::~money_put()(GLIBCXX_3.4) [ISOCXX]
money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::~money_put()(GLIBCXX_3.4) [ISOCXX]

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 9-393, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-393 libstdcxx - Class `money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >` Data Interfaces**

guard variable for <code>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::id(GLIBCXX_3.4)</code> [CXXABI]
<code>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::id(GLIBCXX_3.4)</code> [ISOCXX]
typeid for <code>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
typeid name for <code>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]
vtable for <code>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code> (GLIBCXX_3.4) [CXXABI]

## 9.1.134 Class locale

### 9.1.134.1 Interfaces for Class locale

An LSB conforming implementation shall provide the generic methods for Class `std::locale` specified in Table 9-394, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-394 libstdcxx - Class locale Function Interfaces**

<code>locale::id::_M_id() const(GLIBCXX_3.4)</code> [ISOCXX]
<code>locale::name() const(GLIBCXX_3.4)</code> [ISOCXX]
<code>locale::operator==(locale const&amp;) const(GLIBCXX_3.4)</code> [ISOCXX]
<code>locale::_M_coalesce(locale const&amp;, locale const&amp;, int)(GLIBCXX_3.4)</code> [ISOCXX]
<code>locale::_S_normalize_category(int)(GLIBCXX_3.4)</code> [ISOCXX]
<code>locale::_Impl::_M_install_facet(locale::id const*, locale::facet const*)(GLIBCXX_3.4)</code> [LSB]
<code>locale::_Impl::_M_replace_facet(locale::_Impl const*, locale::id const*)(GLIBCXX_3.4)</code> [LSB]



locale::_Impl::~_Impl()(GLIBCXX_3.4) [LSB]
locale::_Impl::~_Impl()(GLIBCXX_3.4) [LSB]
locale::global(locale const&)(GLIBCXX_3.4) [ISOCXX]
locale::classic()(GLIBCXX_3.4) [ISOCXX]
locale::locale(char const*)(GLIBCXX_3.4) [ISOCXX]
locale::locale(locale::_Impl*)(GLIBCXX_3.4) [ISOCXX]
locale::locale(locale const&)(GLIBCXX_3.4) [ISOCXX]
locale::locale(locale const&, locale const&, int)(GLIBCXX_3.4) [ISOCXX]
locale::locale()(GLIBCXX_3.4) [ISOCXX]
locale::locale(char const*)(GLIBCXX_3.4) [ISOCXX]
locale::locale(locale::_Impl*)(GLIBCXX_3.4) [ISOCXX]
locale::locale(locale const&)(GLIBCXX_3.4) [ISOCXX]
locale::locale(locale const&, char const*, int)(GLIBCXX_3.4) [ISOCXX]
locale::locale(locale const&, locale const&, int)(GLIBCXX_3.4) [ISOCXX]
locale::locale()(GLIBCXX_3.4) [ISOCXX]
locale::~locale()(GLIBCXX_3.4) [ISOCXX]
locale::~locale()(GLIBCXX_3.4) [ISOCXX]
locale::operator=(locale const&)(GLIBCXX_3.4) [ISOCXX]

An LSB conforming implementation shall provide the generic data interfaces for Class `std::locale` specified in Table 9-395, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-395 libstdcxx - Class locale Data Interfaces**

locale::all(GLIBCXX_3.4) [ISOCXX]
locale::none(GLIBCXX_3.4) [ISOCXX]
locale::time(GLIBCXX_3.4) [ISOCXX]
locale::ctype(GLIBCXX_3.4) [ISOCXX]
locale::collate(GLIBCXX_3.4) [ISOCXX]
locale::numeric(GLIBCXX_3.4) [ISOCXX]
locale::messages(GLIBCXX_3.4) [ISOCXX]
locale::monetary(GLIBCXX_3.4) [ISOCXX]

### 9.1.135 Class `locale::facet`

#### 9.1.135.1 Class data for `locale::facet`

The virtual table for the `std::locale::facet` class is described by Table 9-396

**Table 9-396 Primary vtable for locale::facet**

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for locale::facet
vfunc[0]:	locale::facet::~~facet()
vfunc[1]:	locale::facet::~~facet()

The Run Time Type Information for the std::locale::facet class is described by Table 9-397

**Table 9-397 typeinfo for locale::facet**

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for locale::facet

### 9.1.135.2 Interfaces for Class locale::facet

An LSB conforming implementation shall provide the generic methods for Class std::locale::facet specified in Table 9-398, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-398 libstdcxx - Class locale::facet Function Interfaces**

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::facet::~~facet()(GLIBCXX_3.4) [ISOCXX]
locale::facet::~~facet()(GLIBCXX_3.4) [ISOCXX]
locale::facet::~~facet()(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 9-399, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-399 libstdcxx - Class locale::facet Data Interfaces**

__timepunct_cache<char>::_S_timezones(GLIBCXX_3.4) [ISOCXX]
__timepunct_cache<wchar_t>::_S_timezones(GLIBCXX_3.4) [ISOCXX]
typeinfo for locale::facet(GLIBCXX_3.4) [CXXABI]
typeinfo name for locale::facet(GLIBCXX_3.4) [CXXABI]
vtable for locale::facet(GLIBCXX_3.4) [CXXABI]

## 9.1.136 facet functions

### 9.1.136.1 Interfaces for facet functions

An LSB conforming implementation shall provide the generic methods for facet functions specified in Table 9-400, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-400 libstdc++ - facet functions Function Interfaces**

<code>void __convert_to_v&lt;double&gt;(char const*, double&amp;, _Ios_Iostate&amp;, __locale_struct* const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>void __convert_to_v&lt;long double&gt;(char const*, long double&amp;, _Ios_Iostate&amp;, __locale_struct* const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>void __convert_to_v&lt;float&gt;(char const*, float&amp;, _Ios_Iostate&amp;, __locale_struct* const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;money_punct&lt;char, false&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;money_punct&lt;wchar_t, false&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;ctype&lt;wchar_t&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;codecvt&lt;char, char, __mbstate_t&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;codecvt&lt;wchar_t, char, __mbstate_t&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;collate&lt;char&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;collate&lt;wchar_t&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;messages&lt;char&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;messages&lt;wchar_t&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;num_punct&lt;char&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;num_punct&lt;wchar_t&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t,</code>

<code>char_traits&lt;wchar_t&gt; &gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>bool has_facet&lt;money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;&gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>moneypunct&lt;char, false&gt; const&amp; use_facet&lt;moneypunct&lt;char, false&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>moneypunct&lt;char, true&gt; const&amp; use_facet&lt;moneypunct&lt;char, true&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>moneypunct&lt;wchar_t, false&gt; const&amp; use_facet&lt;moneypunct&lt;wchar_t, false&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>moneypunct&lt;wchar_t, true&gt; const&amp; use_facet&lt;moneypunct&lt;wchar_t, true&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct&lt;char&gt; const&amp; use_facet&lt;__timepunct&lt;char&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>__timepunct&lt;wchar_t&gt; const&amp; use_facet&lt;__timepunct&lt;wchar_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>ctype&lt;char&gt; const&amp; use_facet&lt;ctype&lt;char&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>ctype&lt;wchar_t&gt; const&amp; use_facet&lt;ctype&lt;wchar_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt&lt;char, char, __mbstate_t&gt; const&amp; use_facet&lt;codecvt&lt;char, char, __mbstate_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>codecvt&lt;wchar_t, char, __mbstate_t&gt; const&amp; use_facet&lt;codecvt&lt;wchar_t, char, __mbstate_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate&lt;char&gt; const&amp; use_facet&lt;collate&lt;char&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>collate&lt;wchar_t&gt; const&amp; use_facet&lt;collate&lt;wchar_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt; const&amp; use_facet&lt;num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&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;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>

<code>num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; const&amp; use_facet&lt;num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&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;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>messages&lt;char&gt; const&amp; use_facet&lt;messages&lt;char&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>messages&lt;wchar_t&gt; const&amp; use_facet&lt;messages&lt;wchar_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;char&gt; const&amp; use_facet&lt;numpunct&lt;char&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>numpunct&lt;wchar_t&gt; const&amp; use_facet&lt;numpunct&lt;wchar_t&gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; const&amp; use_facet&lt;time_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; const&amp; use_facet&lt;time_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; const&amp; use_facet&lt;time_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; const&amp; use_facet&lt;time_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; const&amp; use_facet&lt;money_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; const&amp; use_facet&lt;money_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; const&amp; use_facet&lt;money_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>
<code>money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; const&amp; use_facet&lt;money_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt; &gt;(locale const&amp;)(GLIBCXX_3.4) [ISOCXX]</code>

### 9.1.137 Class `__num_base`

#### 9.1.137.1 Class data for `__num_base`

The Run Time Type Information for the `std::__num_base` class is described by Table 9-401

Table 9-401

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for __num_base

### 9.1.137.2 Interfaces for Class \_\_num\_base

An LSB conforming implementation shall provide the generic methods for Class `std::__num_base` specified in Table 9-402, with the full mandatory functionality as described in the referenced underlying specification.

Table 9-402 libstdcxx - Class \_\_num\_base Function Interfaces

<code>__num_base::_S_format_float(ios_base const&amp;, char*, char)(GLIBCXX_3.4)</code> [ISOCXX]
---

An LSB conforming implementation shall provide the generic data interfaces for Class `std::__num_base` specified in Table 9-403, with the full mandatory functionality as described in the referenced underlying specification.

Table 9-403 libstdcxx - Class \_\_num\_base Data Interfaces

<code>__num_base::_S_atoms_in(GLIBCXX_3.4)</code> [ISOCXX]
<code>__num_base::_S_atoms_out(GLIBCXX_3.4)</code> [ISOCXX]

### 9.1.138 Class `num_get<char, istreambuf_iterator<char, char_traits<char>>>`

#### 9.1.138.1 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 9-404, with the full mandatory functionality as described in the referenced underlying specification.

Table 9-404 libstdcxx - Class `num_get<char, istreambuf_iterator<char, char_traits<char>>>` Function Interfaces

<code>istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&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)</code> [ISOCXX]
<code>istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&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)</code> [ISOCXX]
<code>istreambuf_iterator&lt;char, char_traits&lt;char&gt;&gt; num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&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;</code>

unsigned long&) const(GLIBCXX_3.4) [ISOCXX]
istreambuf_iterator<char, char_traits<char> > num_get<char, istreambuf_iterator<char, char_traits<char> > >::_M_extract_int<unsigned short>(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, unsigned short&) const(GLIBCXX_3.4) [ISOCXX]
istreambuf_iterator<char, char_traits<char> > num_get<char, istreambuf_iterator<char, char_traits<char> > >::_M_extract_int<long long>(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, long long&) const(GLIBCXX_3.4) [ISOCXX]
istreambuf_iterator<char, char_traits<char> > num_get<char, istreambuf_iterator<char, char_traits<char> > >::_M_extract_int<unsigned long long>(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, unsigned long long&) const(GLIBCXX_3.4) [ISOCXX]
num_get<char, istreambuf_iterator<char, char_traits<char> > >::_M_extract_float(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, ios_base&, _Ios_Iostate&, basic_string<char, char_traits<char>, allocator<char> >&) 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&, void*&) 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&, 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&,</td>





<code>istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;, ios_base&amp;, _Ios_Iostate&amp;, unsigned long&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>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 short&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>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(GLIBCXX_3.4) [ISOCXX]</code>
<code>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(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt;::~~num_get()(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt;::~~num_get()(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt;::~~num_get()(GLIBCXX_3.4) [ISOCXX]</code>

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 9-405, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-405 libstdcxx - Class `num_get<char, istreambuf_iterator<char, char_traits<char> >>` Data Interfaces**

<code>guard variable for num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt;::id(GLIBCXX_3.4) [CXXABI]</code>
<code>num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt;::id(GLIBCXX_3.4) [ISOCXX]</code>
<code>typeid for num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for num_get&lt;char, istreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt;&gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.139 Class `num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> >>`

#### 9.1.139.1 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 9-406, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-406 libstdcxx - Class num\_get<wchar\_t, istreambuf\_iterator<wchar\_t, char\_traits<wchar\_t>>> Function Interfaces**

<pre>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&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;, unsigned int&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&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;, long&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&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;, unsigned long&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&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;, unsigned short&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&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;, long long&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&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;, unsigned long long&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::_M_extract_float(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;, basic_string&lt;char, char_traits&lt;char&gt;, allocator&lt;char&gt; &gt;&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::get(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;, void*&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::get(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;, bool&amp;) const(GLIBCXX_3.4) [ISOCXX]</pre>



<code>_Ios_Iostate&amp;, double&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt; &gt;::do_get(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;, long double&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt; &gt;::do_get(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;, float&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt; &gt;::do_get(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;, unsigned int&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt; &gt;::do_get(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;, long&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt; &gt;::do_get(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;, unsigned long&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt; &gt;::do_get(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;, unsigned short&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt; &gt;::do_get(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;, long long&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt; &gt;::do_get(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;, unsigned long long&amp;) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt; &gt;::~num_get()(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt; &gt;::~num_get()(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_get&lt;wchar_t, istreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt;&gt;&gt; &gt;::~num_get()(GLIBCXX_3.4) [ISOCXX]</code>

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 9-407, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-407 libstdcxx - Class num\_get<wchar\_t, istreambuf\_iterator<wchar\_t, char\_traits<wchar\_t>>> Data Interfaces**

guard variable for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>::id(GLIBCXX_3.4) [CXXABI]
num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>::id(GLIBCXX_3.4) [ISOCXX]
typeid for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>(GLIBCXX_3.4) [CXXABI]
typeid name for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>(GLIBCXX_3.4) [CXXABI]
vtable for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t>>>(GLIBCXX_3.4) [CXXABI]

### 9.1.140 Class num\_put<char, ostreambuf\_iterator<char, char\_traits<char>>>

#### 9.1.140.1 Interfaces for Class num\_put<char, ostreambuf\_iterator<char, char\_traits<char>>>

An LSB conforming implementation shall provide the generic methods for Class std::num\_put<char, std::ostreambuf\_iterator<char, std::char\_traits<char>>> specified in Table 9-408, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-408 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>>>::_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 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



<code>long long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_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, unsigned long long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::~num_put()(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::~num_put()(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::~num_put()(GLIBCXX_3.4) [ISOCXX]</code>

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 9-409, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-409 libstdcxx - Class `num_put<char, ostreambuf_iterator<char, char_traits<char> > >` Data Interfaces**

<code>guard variable for num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::id(GLIBCXX_3.4) [CXXABI]</code>
<code>num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;::id(GLIBCXX_3.4) [ISOCXX]</code>
<code>typeid for num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for num_put&lt;char, ostreambuf_iterator&lt;char, char_traits&lt;char&gt; &gt; &gt;(GLIBCXX_3.4) [CXXABI]</code>

### 9.1.141 Class `num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >`

#### 9.1.141.1 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> > >` specified in Table 9-410, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-410 libstdcxx - Class `num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >` Function Interfaces**

<code>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&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]</code>
<code>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;</code>

<pre>&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]</pre>
<pre>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::_M_insert_int&lt;long long&gt;(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, wchar_t, long long) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; &gt;::_M_insert_int&lt;unsigned long long&gt;(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, wchar_t, unsigned long long) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&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]</pre>
<pre>ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt; num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&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]</pre>
<pre>num_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;, ios_base&amp;, wchar_t, void const*) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>num_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;, ios_base&amp;, wchar_t, bool) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>num_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;, ios_base&amp;, wchar_t, double) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>num_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;, ios_base&amp;, wchar_t, long double) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>num_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;, ios_base&amp;, wchar_t, long) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>num_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;, ios_base&amp;, wchar_t, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>num_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;, ios_base&amp;, wchar_t, long long) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>num_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;, ios_base&amp;, wchar_t, unsigned long long) const(GLIBCXX_3.4) [ISOCXX]</pre>
<pre>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;</pre>



<code>&gt;::do_put(ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;, ios_base&amp;, wchar_t, void const*) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_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;, ios_base&amp;, wchar_t, bool) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_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;, ios_base&amp;, wchar_t, double) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_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;, ios_base&amp;, wchar_t, long double) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_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;, ios_base&amp;, wchar_t, long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_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;, ios_base&amp;, wchar_t, unsigned long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_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;, ios_base&amp;, wchar_t, long long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_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;, ios_base&amp;, wchar_t, unsigned long long) const(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::~~num_put()(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::~~num_put()(GLIBCXX_3.4) [ISOCXX]</code>
<code>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::~~num_put()(GLIBCXX_3.4) [ISOCXX]</code>

An LSB conforming implementation shall provide the generic data interfaces for Class `std::num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> >>` specified in Table 9-411, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-411 libstdcxx - Class `num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> >>` Data Interfaces**

<code>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]</code>
<code>num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;::id(GLIBCXX_3.4) [ISOCXX]</code>
<code>typeid for num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>typeid name for num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t, char_traits&lt;wchar_t&gt; &gt;&gt;(GLIBCXX_3.4) [CXXABI]</code>
<code>vtable for num_put&lt;wchar_t, ostreambuf_iterator&lt;wchar_t,</code>

char_traits<wchar_t> > >(GLIBCXX_3.4) [CXXABI]
--

### 9.1.142 Class `gslice`

#### 9.1.142.1 Class data for `gslice`

#### 9.1.142.2 Interfaces for Class `gslice`

No external methods are defined for `libstdc++` - Class `std::gslice` in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

### 9.1.143 Class `__basic_file<char>`

#### 9.1.143.1 Class data for `__basic_file<char>`

#### 9.1.143.2 Interfaces for Class `__basic_file<char>`

An LSB conforming implementation shall provide the generic methods for Class `std::__basic_file<char>` specified in Table 9-412, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-412 `libstdc++` - Class `__basic_file<char>` Function Interfaces**

<code>__basic_file&lt;char&gt;::is_open() const</code> (GLIBCXX_3.4) [ISOCXX]
<code>__basic_file&lt;char&gt;::fd()</code> (GLIBCXX_3.4) [ISOCXX]
<code>__basic_file&lt;char&gt;::file()</code> (GLIBCXX_3.4.1) [ISOCXX]
<code>__basic_file&lt;char&gt;::open(char const*, _Ios_Openmode, int)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__basic_file&lt;char&gt;::sync()</code> (GLIBCXX_3.4) [ISOCXX]
<code>__basic_file&lt;char&gt;::close()</code> (GLIBCXX_3.4) [ISOCXX]
<code>__basic_file&lt;char&gt;::sys_open(_IO_FILE*, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__basic_file&lt;char&gt;::sys_open(int, _Ios_Openmode)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__basic_file&lt;char&gt;::showmanyc()</code> (GLIBCXX_3.4) [ISOCXX]
<code>__basic_file&lt;char&gt;::__basic_file(pthread_mutex_t*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__basic_file&lt;char&gt;::__basic_file(pthread_mutex_t*)</code> (GLIBCXX_3.4) [ISOCXX]
<code>__basic_file&lt;char&gt;::~~__basic_file()</code> (GLIBCXX_3.4) [ISOCXX]
<code>__basic_file&lt;char&gt;::~~__basic_file()</code> (GLIBCXX_3.4) [ISOCXX]

### 9.1.144 Class `_List_node_base`

#### 9.1.144.1 Interfaces for Class `_List_node_base`

An LSB conforming implementation shall provide the generic methods for Class `std::_List_node_base` specified in Table 9-413, with the full mandatory functionality as described in the referenced underlying specification.

**Table 9-413 libstdcxx - Class `_List_node_base` Function Interfaces**

<code>_List_node_base::hook(_List_node_base*)(GLIBCXX_3.4) [LSB]</code>
<code>_List_node_base::swap(_List_node_base&amp;, _List_node_base&amp;)(GLIBCXX_3.4) [LSB]</code>
<code>_List_node_base::unhook()(GLIBCXX_3.4) [LSB]</code>
<code>_List_node_base::reverse()(GLIBCXX_3.4) [LSB]</code>
<code>_List_node_base::transfer(_List_node_base*, _List_node_base*)(GLIBCXX_3.4) [LSB]</code>

**9.1.145 Class `valarray<unsigned int>`****9.1.145.1 Class data for `valarray<unsigned int>`****9.1.145.2 Interfaces for Class `valarray<unsigned int>`**

No external methods are defined for libstdcxx - Class `std::valarray<unsigned int>` in this part of the specification. See also the relevant architecture specific part of ISO/IEC 23360.

**9.2 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 9.1 shall behave as described in the referenced base document.

## Annex A 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.

### A.1 PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

### A.2 APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

### **A.3 VERBATIM COPYING**

You may copy and distribute the Document in any medium, either commercially or noncommercially, 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.

### **A.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.

## A.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.

## **A.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."

## **A.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.

## **A.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.

## **A.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.

## **A.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.

## A.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.

## A.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.