Linux Standard Base C++ Specification for IA32 4.1

Linux Standard Base C++ Specification for IA32 4.1

Contents

List of Tables

Foreword

This is version 4.1 of the Linux Standard Base C++ Specification for IA32. This specification is one of a series of volumes under the collective title *Linux Standard Base*:

- Core
- C++
- Desktop
- Languages
- Printing

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

Status of this Document

This is a released specification. Other documents may supersede or augment this specification. A list of current Linux Standard Base (LSB) specifications is available at http://refspecs.linuxfoundation.org (http://refspecs.linuxfoundation.org/).

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

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

Introduction

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

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

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

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

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

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

I Introductory Elements

1 Scope

1.1 General

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

These specifications are composed of two basic parts: A common 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") 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	http://www.linuxbase.org/spec/
ISO C (1999)	ISO/IEC 9899: 1999, Programming Languages --C	
ISO/IEC 14882: 2003 C++ Language	ISO/IEC 14882: 2003 Programming languages --C++	
Itanium™ C++ ABI	Itanium™ C++ ABI (Revision 1.86)	http://refspecs.linuxfoundation.org/cxxabi-1.86.html
POSIX 1003.1-2001 (ISO/IEC 9945-2003)	ISO/IEC 9945-1:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 1: Base Definitions ISO/IEC 9945-2:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 2: System Interfaces ISO/IEC 9945-3:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 3: Shell and Utilities ISO/IEC 9945-4:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 4: Rationale Including Technical Cor. 1: 2004	http://www.unix.org/version3/

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 Terms and Definitions

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

archLSB

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

Binary Standard, ABI

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

Implementation-defined

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

Shell Script

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

Source Standard, API

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

Undefined

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

Unspecified

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

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

5 Documentation Conventions

Throughout this document, the following typographic conventions are used:

`function()`

the name of a function

command

the name of a command or utility

`CONSTANT`

a constant value

`parameter`

a parameter

`variable`

a variable

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

name

the name of the interface

(symver)

An optional symbol version identifier, if required.

[*`refno`*]

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

For example,

forkpty(GLIBC_2.0) [SUSv3]

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

> **Note:** For symbols with versions which differ between architectures, the symbol versions are defined in the architecture specific parts of ISO/IEC 23360 only.

II Base Libraries

6 Libraries

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

Interfaces that are unique to the IA32 platform are defined here. This section should be used in conjunction with the corresponding section in the Linux Standard Base Specification.

6.1 Interfaces for libstdcxx

Table 6-1 defines the library name and shared object name for the libstdcxx library

Table 6-1 libstdcxx Definition

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

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

[CXXABI-1.86] Itanium™ C++ ABI
[ISOCXX] ISO/IEC 14882: 2003 C++ Language
[LSB] ISO/IEC 23360 Part 1

6.1.1 C++ Runtime Support

6.1.1.1 Interfaces for C++ Runtime Support

An LSB conforming implementation shall provide the architecture specific methods for C++ Runtime Support specified in Table 6-2, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-2 libstdcxx - C++ Runtime Support Function Interfaces

operator new[](unsigned int)(GLIBCXX_3.4) [ISOCXX]
operator new[](unsigned int, nothrow_t const&)(GLIBCXX_3.4) [ISOCXX]
operator new(unsigned int)(GLIBCXX_3.4) [ISOCXX]
operator new(unsigned int, nothrow_t const&)(GLIBCXX_3.4) [ISOCXX]

6.1.2 C++ type descriptors for built-in types

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

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

6.1.3 C++ _Rb_tree

6.1.3.1 Interfaces for C++ _Rb_tree

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

6.1.4 Class type_info

6.1.4.1 Class data for type_info

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

The Run Time Type Information for the std::type_info class is described by Table 6-3

Table 6-3 typeinfo for type_info

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for type_info

6.1.4.2 Interfaces for Class type_info

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

6.1.5 Class __cxxabiv1::__enum_type_info

6.1.5.1 Class data for __cxxabiv1::__enum_type_info

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

The Run Time Type Information for the __cxxabiv1::__enum_type_info class is described by Table 6-4

Table 6-4 typeinfo for __cxxabiv1::__enum_type_info

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

6.1.5.2 Interfaces for Class __cxxabiv1::__enum_type_info

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

6.1.6 Class __cxxabiv1::__array_type_info

6.1.6.1 Class data for __cxxabiv1::__array_type_info

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

The Run Time Type Information for the __cxxabiv1::__array_type_info class is described by Table 6-5

Table 6-5 typeinfo for __cxxabiv1::__array_type_info

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

	__cxxabiv1::__array_type_info

6.1.6.2 Interfaces for Class __cxxabiv1::__array_type_info

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

6.1.7 Class __cxxabiv1::__class_type_info

6.1.7.1 Class data for __cxxabiv1::__class_type_info

The virtual table for the __cxxabiv1::__class_type_info class is described by Table 6-6

Table 6-6 Primary vtable for __cxxabiv1::__class_type_info

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for __cxxabiv1::__class_type_info
vfunc[0]:	__cxxabiv1::__class_type_info::~__class_type_info()
vfunc[1]:	__cxxabiv1::__class_type_info::~__class_type_info()
vfunc[2]:	type_info::__is_pointer_p() const
vfunc[3]:	type_info::__is_function_p() const
vfunc[4]:	__cxxabiv1::__class_type_info::__do_catch(type_info const*, void**, unsigned int) const
vfunc[5]:	__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const
vfunc[6]:	__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&) const
vfunc[7]:	__cxxabiv1::__class_type_info::__do_dyncast(int, __cxxabiv1::__class_type_info::__sub_kind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__dyncast_result&) const
vfunc[8]:	__cxxabiv1::__class_type_info::__do_find_public_src(int, void const*, __cxxabiv1::__class_type_info const*,

	void const*) const

The Run Time Type Information for the __cxxabiv1::__class_type_info class is described by Table 6-7

Table 6-7 typeinfo for __cxxabiv1::__class_type_info

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

6.1.7.2 Interfaces for Class __cxxabiv1::__class_type_info

An LSB conforming implementation shall provide the architecture specific methods for Class __cxxabiv1::__class_type_info specified in Table 6-8, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-8 libstdcxx - Class __cxxabiv1::__class_type_info Function Interfaces

__cxxabiv1::__class_type_info::__do_dyncast(int, __cxxabiv1::__class_type_info::__sub_kind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__dyncast_result&) const(CXXABI_1.3) [CXXABI-1.86]
__cxxabiv1::__class_type_info::__do_find_public_src(int, void const*, __cxxabiv1::__class_type_info const*, void const*) const(CXXABI_1.3) [CXXABI-1.86]

6.1.8 Class __cxxabiv1::__pbase_type_info

6.1.8.1 Class data for __cxxabiv1::__pbase_type_info

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

The Run Time Type Information for the __cxxabiv1::__pbase_type_info class is described by Table 6-9

Table 6-9 typeinfo for __cxxabiv1::__pbase_type_info

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

6.1.8.2 Interfaces for Class __cxxabiv1::__pbase_type_info

No external methods are defined for libstdcxx - Class __cxxabiv1::__pbase_type_info in this part of the specification. See also the generic specification.

6.1.9 Class __cxxabiv1::__pointer_type_info

6.1.9.1 Class data for __cxxabiv1::__pointer_type_info

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

The Run Time Type Information for the __cxxabiv1::__pointer_type_info class is described by Table 6-10

Table 6-10 typeinfo for __cxxabiv1::__pointer_type_info

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

6.1.9.2 Interfaces for Class __cxxabiv1::__pointer_type_info

No external methods are defined for libstdcxx - Class __cxxabiv1::__pointer_type_info in this part of the specification. See also the generic specification.

6.1.10 Class __cxxabiv1::__function_type_info

6.1.10.1 Class data for __cxxabiv1::__function_type_info

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

The Run Time Type Information for the __cxxabiv1::__function_type_info class is described by Table 6-11

Table 6-11 typeinfo for __cxxabiv1::__function_type_info

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

6.1.10.2 Interfaces for Class __cxxabiv1::__function_type_info

No external methods are defined for libstdcxx - Class __cxxabiv1::__function_type_info in this part of the specification. See also the generic specification.

6.1.11 Class __cxxabiv1::__si_class_type_info

6.1.11.1 Class data for __cxxabiv1::__si_class_type_info

The virtual table for the __cxxabiv1::__si_class_type_info class is described by Table 6-12

Table 6-12 Primary vtable for __cxxabiv1::__si_class_type_info

Base Offset	0
Virtual Base Offset	0

RTTI	typeinfo for __cxxabiv1::__si_class_type_info
vfunc[0]:	__cxxabiv1::__si_class_type_info::~__si_class_type_info()
vfunc[1]:	__cxxabiv1::__si_class_type_info::~__si_class_type_info()
vfunc[2]:	type_info::__is_pointer_p() const
vfunc[3]:	type_info::__is_function_p() const
vfunc[4]:	__cxxabiv1::__class_type_info::__do_catch(type_info const*, void**, unsigned int) const
vfunc[5]:	__cxxabiv1::__class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void**) const
vfunc[6]:	__cxxabiv1::__si_class_type_info::__do_upcast(__cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__upcast_result&) const
vfunc[7]:	__cxxabiv1::__si_class_type_info::__do_dyncast(int, __cxxabiv1::__class_type_info::__sub_kind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__dyncast_result&) const
vfunc[8]:	__cxxabiv1::__si_class_type_info::__do_find_public_src(int, void const*, __cxxabiv1::__class_type_info const*, void const*) const

The Run Time Type Information for the __cxxabiv1::__si_class_type_info class is described by Table 6-13

Table 6-13 typeinfo for __cxxabiv1::__si_class_type_info

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

6.1.11.2 Interfaces for Class __cxxabiv1::__si_class_type_info

An LSB conforming implementation shall provide the architecture specific methods for Class __cxxabiv1::__si_class_type_info specified in Table 6-14, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-14 libstdcxx - Class __cxxabiv1::__si_class_type_info Function Interfaces

__cxxabiv1::__si_class_type_info::__do_dyncast(int, __cxxabiv1::__class_type_info::__sub_kind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__dyncast_result&) const(CXXABI_1.3) [CXXABI-1.86]
__cxxabiv1::__si_class_type_info::__do_find_public_src(int, void const*, __cxxabiv1::__class_type_info const*, void const*) const(CXXABI_1.3) [CXXABI-1.86]

6.1.12 Class __cxxabiv1::__vmi_class_type_info

6.1.12.1 Class data for __cxxabiv1::__vmi_class_type_info

The virtual table for the __cxxabiv1::__vmi_class_type_info class is described by Table 6-15

Table 6-15 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::__upcast_result&) const
vfunc[7]:	__cxxabiv1::__vmi_class_type_info::__do_dyncast(int, __cxxabiv1::__class_type_info::__sub_kind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*,

	__cxxabiv1::__class_type_info::__dyncast_result&) const
vfunc[8]:	__cxxabiv1::__vmi_class_type_info::__do_find_public_src(int, void const*, __cxxabiv1::__class_type_info const*, void const*) const

The Run Time Type Information for the __cxxabiv1::__vmi_class_type_info class is described by Table 6-16

Table 6-16 typeinfo for __cxxabiv1::__vmi_class_type_info

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

6.1.12.2 Interfaces for Class __cxxabiv1::__vmi_class_type_info

An LSB conforming implementation shall provide the architecture specific methods for Class __cxxabiv1::__vmi_class_type_info specified in Table 6-17, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-17 libstdcxx - Class __cxxabiv1::__vmi_class_type_info Function Interfaces

__cxxabiv1::__vmi_class_type_info::__do_dyncast(int, __cxxabiv1::__class_type_info::__sub_kind, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info const*, void const*, __cxxabiv1::__class_type_info::__dyncast_result&) const(CXXABI_1.3) [CXXABI-1.86]
__cxxabiv1::__vmi_class_type_info::__do_find_public_src(int, void const*, __cxxabiv1::__class_type_info const*, void const*) const(CXXABI_1.3) [CXXABI-1.86]

6.1.13 Class __cxxabiv1::__fundamental_type_info

6.1.13.1 Class data for __cxxabiv1::__fundamental_type_info

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

The Run Time Type Information for the __cxxabiv1::__fundamental_type_info class is described by Table 6-18

Table 6-18 typeinfo for __cxxabiv1::__fundamental_type_info

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

6.1.13.2 Interfaces for Class __cxxabiv1::__fundamental_type_info

No external methods are defined for libstdcxx - Class __cxxabiv1::__fundamental_type_info in this part of the specification. See also the generic specification.

6.1.14 Class __cxxabiv1::__pointer_to_member_type_info

6.1.14.1 Class data for __cxxabiv1::__pointer_to_member_type_info

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

The Run Time Type Information for the __cxxabiv1::__pointer_to_member_type_info class is described by Table 6-19

Table 6-19 typeinfo for __cxxabiv1::__pointer_to_member_type_info

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

6.1.14.2 Interfaces for Class __cxxabiv1::__pointer_to_member_type_info

No external methods are defined for libstdcxx - Class __cxxabiv1::__pointer_to_member_type_info in this part of the specification. See also the generic specification.

6.1.15 Class __gnu_cxx::stdio_filebuf<char, char_traits<char> >

6.1.15.1 Interfaces for Class __gnu_cxx::stdio_filebuf<char, char_traits<char> >

No external methods are defined for libstdcxx - Class __gnu_cxx::stdio_filebuf<char, std::char_traits<char> > in this part of the specification. See also the generic specification.

6.1.16 Class __gnu_cxx::stdio_filebuf<wchar_t, char_traits<wchar_t> >

6.1.16.1 Interfaces for Class __gnu_cxx::stdio_filebuf<wchar_t, char_traits<wchar_t> >

No external methods are defined for libstdcxx - Class __gnu_cxx::stdio_filebuf<wchar_t, std::char_traits<wchar_t> > in this part of the specification. See also the generic specification.

6.1.17 Class __gnu_cxx::__pool_alloc_base

6.1.17.1 Interfaces for Class __gnu_cxx::__pool_alloc_base

An LSB conforming implementation shall provide the architecture specific methods for Class __gnu_cxx::__pool_alloc_base specified in Table 6-20, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-20 libstdcxx - Class __gnu_cxx::__pool_alloc_base Function Interfaces

__gnu_cxx::__pool_alloc_base::_M_get_free_list(unsigned int)(GLIBCXX_3.4.2) [LSB]
__gnu_cxx::__pool_alloc_base::_M_refill(unsigned int)(GLIBCXX_3.4.2) [LSB]

6.1.18 Class __gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >

6.1.18.1 Class data for __gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >

The virtual table for the __gnu_cxx::stdio_sync_filebuf<char, std::char_traits<char> > class is described by Table 6-21

Table 6-21 Primary vtable for __gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for __gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >
vfunc[0]:	__gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >::~stdio_sync_filebuf()
vfunc[1]:	__gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >::~stdio_sync_filebuf()
vfunc[2]:	basic_streambuf<char, char_traits<char> >::imbue(locale const&)
vfunc[3]:	basic_streambuf<char, char_traits<char> >::setbuf(char*, int)
vfunc[4]:	__gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)
vfunc[5]:	__gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	__gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >::sync()
vfunc[7]:	basic_streambuf<char, char_traits<char> >::showmanyc()
vfunc[8]:	__gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >::xsgetn(char*, int)

vfunc[9]:	__gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >::underflow()
vfunc[10]:	__gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >::uflow()
vfunc[11]:	__gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >::pbackfail(int)
vfunc[12]:	__gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >::xsputn(char const*, int)
vfunc[13]:	__gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >::overflow(int)

6.1.18.2 Interfaces for Class __gnu_cxx::stdio_sync_filebuf<char, char_traits<char> >

No external methods are defined for libstdcxx - Class __gnu_cxx::stdio_sync_filebuf<char, std::char_traits<char> > in this part of the specification. See also the generic specification.

6.1.19 Class __gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >

6.1.19.1 Class data for __gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >

The virtual table for the __gnu_cxx::stdio_sync_filebuf<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-22

Table 6-22 Primary vtable for __gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for __gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >
vfunc[0]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::~stdio_sync_filebuf()
vfunc[1]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::~stdio_sync_filebuf()
vfunc[2]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::imbue(locale const&)
vfunc[3]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::setbuf(wchar_t*, int)
vfunc[4]:	__gnu_cxx::stdio_sync_filebuf<wcha

	r_t, char_traits<wchar_t> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)
vfunc[5]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::sync()
vfunc[7]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::showmanyc()
vfunc[8]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::xsgetn(wchar_t*, int)
vfunc[9]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::underflow()
vfunc[10]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::uflow()
vfunc[11]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::pbackfail(unsigned int)
vfunc[12]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::xsputn(wchar_t const*, int)
vfunc[13]:	__gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >::overflow(unsigned int)

6.1.19.2 Interfaces for Class __gnu_cxx::stdio_sync_filebuf<wchar_t, char_traits<wchar_t> >

No external methods are defined for libstdcxx - Class __gnu_cxx::stdio_sync_filebuf<wchar_t, std::char_traits<wchar_t> > in this part of the specification. See also the generic specification.

6.1.20 Class exception

6.1.20.1 Class data for exception

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

The Run Time Type Information for the std::exception class is described by Table 6-23

Table 6-23 typeinfo for exception

Base Vtable	vtable for __cxxabiv1::__class_type_info

Name	typeinfo name for exception

6.1.20.2 Interfaces for Class exception

No external methods are defined for libstdcxx - Class std::exception in this part of the specification. See also the generic specification.

6.1.21 Class bad_typeid

6.1.21.1 Class data for bad_typeid

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

The Run Time Type Information for the std::bad_typeid class is described by Table 6-24

Table 6-24 typeinfo for bad_typeid

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

6.1.21.2 Interfaces for Class bad_typeid

No external methods are defined for libstdcxx - Class std::bad_typeid in this part of the specification. See also the generic specification.

6.1.22 Class logic_error

6.1.22.1 Class data for logic_error

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

The Run Time Type Information for the std::logic_error class is described by Table 6-25

Table 6-25 typeinfo for logic_error

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

6.1.22.2 Interfaces for Class logic_error

No external methods are defined for libstdcxx - Class std::logic_error in this part of the specification. See also the generic specification.

6.1.23 Class range_error

6.1.23.1 Class data for range_error

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

The Run Time Type Information for the std::range_error class is described by Table 6-26

Table 6-26 typeinfo for range_error

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

6.1.23.2 Interfaces for Class range_error

No external methods are defined for libstdcxx - Class std::range_error in this part of the specification. See also the generic specification.

6.1.24 Class domain_error

6.1.24.1 Class data for domain_error

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

The Run Time Type Information for the std::domain_error class is described by Table 6-27

Table 6-27 typeinfo for domain_error

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

6.1.24.2 Interfaces for Class domain_error

No external methods are defined for libstdcxx - Class std::domain_error in this part of the specification. See also the generic specification.

6.1.25 Class length_error

6.1.25.1 Class data for length_error

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

The Run Time Type Information for the std::length_error class is described by Table 6-28

Table 6-28 typeinfo for length_error

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

6.1.25.2 Interfaces for Class length_error

No external methods are defined for libstdcxx - Class std::length_error in this part of the specification. See also the generic specification.

6.1.26 Class out_of_range

6.1.26.1 Class data for out_of_range

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

The Run Time Type Information for the std::out_of_range class is described by Table 6-29

Table 6-29 typeinfo for out_of_range

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

6.1.26.2 Interfaces for Class out_of_range

No external methods are defined for libstdcxx - Class std::out_of_range in this part of the specification. See also the generic specification.

6.1.27 Class bad_exception

6.1.27.1 Class data for bad_exception

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

The Run Time Type Information for the std::bad_exception class is described by Table 6-30

Table 6-30 typeinfo for bad_exception

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

6.1.27.2 Interfaces for Class bad_exception

No external methods are defined for libstdcxx - Class std::bad_exception in this part of the specification. See also the generic specification.

6.1.28 Class runtime_error

6.1.28.1 Class data for runtime_error

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

The Run Time Type Information for the std::runtime_error class is described by Table 6-31

Table 6-31 typeinfo for runtime_error

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

6.1.28.2 Interfaces for Class runtime_error

No external methods are defined for libstdcxx - Class std::runtime_error in this part of the specification. See also the generic specification.

6.1.29 Class overflow_error

6.1.29.1 Class data for overflow_error

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

The Run Time Type Information for the std::overflow_error class is described by Table 6-32

Table 6-32 typeinfo for overflow_error

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

6.1.29.2 Interfaces for Class overflow_error

No external methods are defined for libstdcxx - Class std::overflow_error in this part of the specification. See also the generic specification.

6.1.30 Class underflow_error

6.1.30.1 Class data for underflow_error

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

The Run Time Type Information for the std::underflow_error class is described by Table 6-33

Table 6-33 typeinfo for underflow_error

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

6.1.30.2 Interfaces for Class underflow_error

No external methods are defined for libstdcxx - Class std::underflow_error in this part of the specification. See also the generic specification.

6.1.31 Class invalid_argument

6.1.31.1 Class data for invalid_argument

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

The Run Time Type Information for the std::invalid_argument class is described by Table 6-34

Table 6-34 typeinfo for invalid_argument

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

6.1.31.2 Interfaces for Class invalid_argument

No external methods are defined for libstdcxx - Class std::invalid_argument in this part of the specification. See also the generic specification.

6.1.32 Class bad_cast

6.1.32.1 Class data for bad_cast

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

The Run Time Type Information for the std::bad_cast class is described by Table 6-35

Table 6-35 typeinfo for bad_cast

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

6.1.32.2 Interfaces for Class bad_cast

No external methods are defined for libstdcxx - Class std::bad_cast in this part of the specification. See also the generic specification.

6.1.33 Class bad_alloc

6.1.33.1 Class data for bad_alloc

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

The Run Time Type Information for the std::bad_alloc class is described by Table 6-36

Table 6-36 typeinfo for bad_alloc

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

6.1.33.2 Interfaces for Class bad_alloc

No external methods are defined for libstdcxx - Class std::bad_alloc in this part of the specification. See also the generic specification.

6.1.34 struct __numeric_limits_base

6.1.34.1 Interfaces for struct __numeric_limits_base

No external methods are defined for libstdcxx - struct __numeric_limits_base in this part of the specification. See also the generic specification.

6.1.35 struct numeric_limits<long double>

6.1.35.1 Interfaces for struct numeric_limits<long double>

No external methods are defined for libstdcxx - struct numeric_limits<long double> in this part of the specification. See also the generic specification.

6.1.36 struct numeric_limits<long long>

6.1.36.1 Interfaces for struct numeric_limits<long long>

No external methods are defined for libstdcxx - struct numeric_limits<long long> in this part of the specification. See also the generic specification.

6.1.37 struct numeric_limits<unsigned long long>

6.1.37.1 Interfaces for struct numeric_limits<unsigned long long>

No external methods are defined for libstdcxx - struct numeric_limits<unsigned long long> in this part of the specification. See also the generic specification.

6.1.38 struct numeric_limits<float>

6.1.38.1 Interfaces for struct numeric_limits<float>

No external methods are defined for libstdcxx - struct numeric_limits<float> in this part of the specification. See also the generic specification.

6.1.39 struct numeric_limits<double>

6.1.39.1 Interfaces for struct numeric_limits<double>

No external methods are defined for libstdcxx - struct numeric_limits<double> in this part of the specification. See also the generic specification.

6.1.40 struct numeric_limits<short>

6.1.40.1 Interfaces for struct numeric_limits<short>

No external methods are defined for libstdcxx - struct numeric_limits<short> in this part of the specification. See also the generic specification.

6.1.41 struct numeric_limits<unsigned short>

6.1.41.1 Interfaces for struct numeric_limits<unsigned short>

No external methods are defined for libstdcxx - struct numeric_limits<unsigned short> in this part of the specification. See also the generic specification.

6.1.42 struct numeric_limits<int>

6.1.42.1 Interfaces for struct numeric_limits<int>

No external methods are defined for libstdcxx - struct numeric_limits<int> in this part of the specification. See also the generic specification.

6.1.43 struct numeric_limits<unsigned int>

6.1.43.1 Interfaces for struct numeric_limits<unsigned int>

No external methods are defined for libstdcxx - struct numeric_limits<unsigned int> in this part of the specification. See also the generic specification.

6.1.44 struct numeric_limits<long>

6.1.44.1 Interfaces for struct numeric_limits<long>

No external methods are defined for libstdcxx - struct numeric_limits<long> in this part of the specification. See also the generic specification.

6.1.45 struct numeric_limits<unsigned long>

6.1.45.1 Interfaces for struct numeric_limits<unsigned long>

No external methods are defined for libstdcxx - struct numeric_limits<unsigned long> in this part of the specification. See also the generic specification.

6.1.46 struct numeric_limits<wchar_t>

6.1.46.1 Interfaces for struct numeric_limits<wchar_t>

No external methods are defined for libstdcxx - struct numeric_limits<wchar_t> in this part of the specification. See also the generic specification.

6.1.47 struct numeric_limits<unsigned char>

6.1.47.1 Interfaces for struct numeric_limits<unsigned char>

No external methods are defined for libstdcxx - struct numeric_limits<unsigned char> in this part of the specification. See also the generic specification.

6.1.48 struct numeric_limits<signed char>

6.1.48.1 Interfaces for struct numeric_limits<signed char>

No external methods are defined for libstdcxx - struct numeric_limits<signed char> in this part of the specification. See also the generic specification.

6.1.49 struct numeric_limits<char>

6.1.49.1 Interfaces for struct numeric_limits<char>

No external methods are defined for libstdcxx - struct numeric_limits<char> in this part of the specification. See also the generic specification.

6.1.50 struct numeric_limits<bool>

6.1.50.1 Interfaces for struct numeric_limits<bool>

No external methods are defined for libstdcxx - struct numeric_limits<bool> in this part of the specification. See also the generic specification.

6.1.51 Class ctype_base

6.1.51.1 Class data for ctype_base

The Run Time Type Information for the std::ctype_base class is described by Table 6-37

Table 6-37 typeinfo for ctype_base

Base Vtable	vtable for __cxxabiv1::__class_type_info

Name	typeinfo name for ctype_base

6.1.51.2 Interfaces for Class ctype_base

No external methods are defined for libstdcxx - Class std::ctype_base in this part of the specification. See also the generic specification.

6.1.52 Class __ctype_abstract_base<char>

6.1.52.1 Class data for __ctype_abstract_base<char>

The virtual table for the std::__ctype_abstract_base<char> class is described in the generic part of this specification.

6.1.52.2 Interfaces for Class __ctype_abstract_base<char>

No external methods are defined for libstdcxx - Class std::__ctype_abstract_base<char> in this part of the specification. See also the generic specification.

6.1.53 Class __ctype_abstract_base<wchar_t>

6.1.53.1 Class data for __ctype_abstract_base<wchar_t>

The virtual table for the std::__ctype_abstract_base<wchar_t> class is described in the generic part of this specification.

6.1.53.2 Interfaces for Class __ctype_abstract_base<wchar_t>

No external methods are defined for libstdcxx - Class std::__ctype_abstract_base<wchar_t> in this part of the specification. See also the generic specification.

6.1.54 Class ctype<char>

6.1.54.1 Class data for ctype<char>

The virtual table for the std::ctype<char> class is described in the generic part of this specification.

6.1.54.2 Interfaces for Class ctype<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::ctype<char> specified in Table 6-38, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-38 libstdcxx - Class ctype<char> Function Interfaces

ctype<char>::ctype(__locale_struct*, unsigned short const*, bool, unsigned int)(GLIBCXX_3.4) [ISOCXX]
ctype<char>::ctype(unsigned short const*, bool, unsigned int)(GLIBCXX_3.4) [ISOCXX]
ctype<char>::ctype(__locale_struct*, unsigned short const*, bool, unsigned int)(GLIBCXX_3.4) [ISOCXX]
ctype<char>::ctype(unsigned short const*, bool, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.55 Class ctype<wchar_t>

6.1.55.1 Class data for ctype<wchar_t>

The virtual table for the std::ctype<wchar_t> class is described in the generic part of this specification.

The Run Time Type Information for the std::ctype<wchar_t> class is described by Table 6-39

Table 6-39 typeinfo for ctype<wchar_t>

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

6.1.55.2 Interfaces for Class ctype<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::ctype<wchar_t> specified in Table 6-40, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-40 libstdcxx - Class ctype<wchar_t> Function Interfaces

ctype<wchar_t>::ctype(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::ctype(unsigned int)(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::ctype(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
ctype<wchar_t>::ctype(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.56 Class ctype_byname<char>

6.1.56.1 Class data for ctype_byname<char>

The virtual table for the std::ctype_byname<char> class is described in the generic part of this specification.

The Run Time Type Information for the std::ctype_byname<char> class is described by Table 6-41

Table 6-41 typeinfo for ctype_byname<char>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for ctype_byname<char>

6.1.56.2 Interfaces for Class ctype_byname<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::ctype_byname<char> specified in Table 6-42, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-42 libstdcxx - Class ctype_byname<char> Function Interfaces

ctype_byname<char>::ctype_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
ctype_byname<char>::ctype_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.57 Class ctype_byname<wchar_t>

6.1.57.1 Class data for ctype_byname<wchar_t>

The virtual table for the std::ctype_byname<wchar_t> class is described in the generic part of this specification.

The Run Time Type Information for the std::ctype_byname<wchar_t> class is described by Table 6-43

Table 6-43 typeinfo for ctype_byname<wchar_t>

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

6.1.57.2 Interfaces for Class ctype_byname<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::ctype_byname<wchar_t> specified in Table 6-44, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-44 libstdcxx - Class ctype_byname<wchar_t> Function Interfaces

ctype_byname<wchar_t>::ctype_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
ctype_byname<wchar_t>::ctype_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.58 Class basic_string<char, char_traits<char>, allocator<char> >

6.1.58.1 Interfaces for Class basic_string<char, char_traits<char>, allocator<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_string<char, std::char_traits<char>, std::allocator<char> > specified in Table 6-45, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-45 libstdcxx - Class basic_string<char, char_traits<char>, allocator<char> > Function Interfaces

basic_string<char, char_traits<char>, allocator<char> >::find_last_of(char const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_last_of(char

const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_last_of(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_last_of(char, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_first_of(char const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_first_of(char const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_first_of(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_first_of(char, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_check_length(unsigned int, unsigned int, char const*) const(GLIBCXX_3.4.5) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_last_not_of(char const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_last_not_of(char const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_last_not_of(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_last_not_of(char, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_first_not_of(char const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_first_not_of(char const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_first_not_of(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find_first_not_of(char, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::at(unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::copy(char*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find(char const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::find(char const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::find(char, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::rfind(char const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::rfind(char const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::rfind(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::rfind(char, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::substr(unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::compare(unsigned int, unsigned int, char const*) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::compare(unsigned int, unsigned int, char const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::compare(unsigned int, unsigned int, basic_string<char, char_traits<char>, allocator<char> > const&) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::compare(unsigned int, unsigned int, basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_check(unsigned int, char const*) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_limit(unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::operator[](unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_S_construct(unsigned int, char, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_replace_aux(unsigned int, unsigned int, unsigned int, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_replace_safe(unsigned int, unsigned int, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::at(unsigned

int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_Rep::_M_set_length_and_sharable(unsigned int)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_Rep::_M_clone(allocator<char> const&, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_Rep::_S_create(unsigned int, unsigned int, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::erase(unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::append(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::append(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::append(unsigned int, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::assign(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::assign(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::assign(unsigned int, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::insert(__gnu_cxx::__normal_iterator<char*, basic_string<char, char_traits<char>, allocator<char> > >, unsigned int, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::insert(unsigned int, char const*)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::insert(unsigned int, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::insert(unsigned int, basic_string<char, char_traits<char>, allocator<char> > const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::insert(unsigned int, basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::insert(unsigned int, unsigned int, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::resize(unsigned int)(GLIBCXX_3.4) [ISOCXX]

basic_string<char, char_traits<char>, allocator<char> >::resize(unsigned int, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_copy(char*, char const*, unsigned int)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_move(char*, char const*, unsigned int)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(__gnu_cxx::__normal_iterator<char*, basic_string<char, char_traits<char>, allocator<char> > >, __gnu_cxx::__normal_iterator<char*, basic_string<char, char_traits<char>, allocator<char> > >, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(__gnu_cxx::__normal_iterator<char*, basic_string<char, char_traits<char>, allocator<char> > >, __gnu_cxx::__normal_iterator<char*, basic_string<char, char_traits<char>, allocator<char> > >, unsigned int, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(unsigned int, unsigned int, char const*)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(unsigned int, unsigned int, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(unsigned int, unsigned int, basic_string<char, char_traits<char>, allocator<char> > const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(unsigned int, unsigned int, basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::replace(unsigned int, unsigned int, unsigned int, char)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::reserve(unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_assign(char*, unsigned int, char)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::_M_mutate(unsigned int, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::basic_string(char const*, unsigned int, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::basic_string(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::basic_string(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int, unsigned int, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char>

>::basic_string(unsigned int, char, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::basic_string(char const*, unsigned int, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::basic_string(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::basic_string(basic_string<char, char_traits<char>, allocator<char> > const&, unsigned int, unsigned int, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::basic_string(unsigned int, char, allocator<char> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<char, char_traits<char>, allocator<char> >::operator[](unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.59 Class basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

6.1.59.1 Interfaces for Class basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_string<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > specified in Table 6-46, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-46 libstdcxx - Class basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > Function Interfaces

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_last_of(wchar_t const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_last_of(wchar_t const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_last_of(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_last_of(wchar_t, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_first_of(wchar_t const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_first_of(wchar_t const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_first_of(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_first_of(wchar_t, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_M_check_length(unsigned int, unsigned int, char const*) const(GLIBCXX_3.4.5) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_last_not_of(wchar_t const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_last_not_of(wchar_t const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_last_not_of(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_last_not_of(wchar_t, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_first_not_of(wchar_t const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_first_not_of(wchar_t const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_first_not_of(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find_first_not_of(wchar_t, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::at(unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::copy(wchar_t*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find(wchar_t const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find(wchar_t const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::find(wchar_t, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::rfind(wchar_t const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t>

>::rfind(wchar_t const*, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::rfind(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::rfind(wchar_t, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::substr(unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::compare(unsigned int, unsigned int, wchar_t const*) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::compare(unsigned int, unsigned int, wchar_t const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::compare(unsigned int, unsigned int, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::compare(unsigned int, unsigned int, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_M_check(unsigned int, char const*) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_M_limit(unsigned int, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::operator[](unsigned int) const(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_S_construct(unsigned int, wchar_t, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_M_replace_aux(unsigned int, unsigned int, unsigned int, wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_M_replace_safe(unsigned int, unsigned int, wchar_t const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::at(unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::_M_set_length_and_sharable(unsigned int)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::_M_clone(allocator<wchar_t> const&, unsigned int)(GLIBCXX_3.4)

[ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_Rep::_S_create(unsigned int, unsigned int, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::erase(unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::append(wchar_t const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::append(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::append(unsigned int, wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::assign(wchar_t const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::assign(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::assign(unsigned int, wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::insert(__gnu_cxx::__normal_iterator<wchar_t*, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > >, unsigned int, wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::insert(unsigned int, wchar_t const*)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::insert(unsigned int, wchar_t const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::insert(unsigned int, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::insert(unsigned int, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::insert(unsigned int, unsigned int, wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::resize(unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::resize(unsigned int, wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_M_copy(wchar_t*, wchar_t const*, unsigned int)(GLIBCXX_3.4.5) [ISOCXX]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_M_move(wchar_t*, wchar_t const*, unsigned int)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::replace(__gnu_cxx::__normal_iterator<wchar_t*, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > >, __gnu_cxx::__normal_iterator<wchar_t*, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > >, wchar_t const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::replace(__gnu_cxx::__normal_iterator<wchar_t*, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > >, __gnu_cxx::__normal_iterator<wchar_t*, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > >, unsigned int, wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::replace(unsigned int, unsigned int, wchar_t const*)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::replace(unsigned int, unsigned int, wchar_t const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::replace(unsigned int, unsigned int, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::replace(unsigned int, unsigned int, basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::replace(unsigned int, unsigned int, unsigned int, wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::reserve(unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_M_assign(wchar_t*, unsigned int, wchar_t)(GLIBCXX_3.4.5) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_M_mutate(unsigned int, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::basic_string(wchar_t const*, unsigned int, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::basic_string(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]

basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::basic_string(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int, unsigned int, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::basic_string(unsigned int, wchar_t, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::basic_string(wchar_t const*, unsigned int, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::basic_string(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::basic_string(basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > const&, unsigned int, unsigned int, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::basic_string(unsigned int, wchar_t, allocator<wchar_t> const&)(GLIBCXX_3.4) [ISOCXX]
basic_string<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::operator[](unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.60 Class basic_stringstream<char, char_traits<char>, allocator<char> >

6.1.60.1 Class data for basic_stringstream<char, char_traits<char>, allocator<char> >

The virtual table for the std::basic_stringstream<char, std::char_traits<char>, std::allocator<char> > class is described by Table 6-47

Table 6-47 Primary vtable for basic_stringstream<char, char_traits<char>, allocator<char> >

Base Offset	0
Virtual Base Offset	52
RTTI	typeinfo for basic_stringstream<char, char_traits<char>, allocator<char> >
vfunc[0]:	basic_stringstream<char, char_traits<char>, allocator<char> >::~basic_stringstream()
vfunc[1]:	basic_stringstream<char, char_traits<char>, allocator<char> >::~basic_stringstream()

Table 6-48 Secondary vtable for basic_stringstream<char, char_traits<char>, allocator<char> >

Base Offset	-8
Virtual Base Offset	44
RTTI	typeinfo for basic_stringstream<char, char_traits<char>, allocator<char> >
vfunc[0]:	non-virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char> >::~basic_stringstream()
vfunc[1]:	non-virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char> >::~basic_stringstream()

Table 6-49 Secondary vtable for basic_stringstream<char, char_traits<char>, allocator<char> >

Base Offset	-52
Virtual Base Offset	-52
RTTI	typeinfo for basic_stringstream<char, char_traits<char>, allocator<char> >
vfunc[0]:	virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char> >::~basic_stringstream()
vfunc[1]:	virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char> >::~basic_stringstream()

The VTT for the std::basic_stringstream<char, std::char_traits<char>, std::allocator<char> > class is described by Table 6-50

Table 6-50 VTT for basic_stringstream<char, char_traits<char>, allocator<char> >

VTT Name	_ZTTSt18basic_stringstreamIcSt11char_traitsIcESaIcEE
Number of Entries	10

6.1.60.2 Interfaces for Class basic_stringstream<char, char_traits<char>, allocator<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_stringstream<char, std::char_traits<char>, std::allocator<char> > specified in Table 6-51, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-51 libstdcxx - Class basic_stringstream<char, char_traits<char>, allocator<char> > Function Interfaces

non-virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char> >::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char> >::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char> >::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_stringstream<char, char_traits<char>, allocator<char> >::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.61 Class basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

6.1.61.1 Class data for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

The virtual table for the std::basic_stringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > class is described by Table 6-52

Table 6-52 Primary vtable for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

Base Offset	0
Virtual Base Offset	52
RTTI	typeinfo for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >
vfunc[0]:	basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringstream()
vfunc[1]:	basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringstream()

Table 6-53 Secondary vtable for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

Base Offset	-8
Virtual Base Offset	44
RTTI	typeinfo for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >
vfunc[0]:	non-virtual thunk to

	basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringstream()
vfunc[1]:	non-virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringstream()

Table 6-54 Secondary vtable for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

Base Offset	-52
Virtual Base Offset	-52
RTTI	typeinfo for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >
vfunc[0]:	virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringstream()
vfunc[1]:	virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringstream()

The VTT for the std::basic_stringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > class is described by Table 6-55

Table 6-55 VTT for basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

VTT Name	_ZTTSt18basic_stringstreamIwSt11char_traitsIwESaIwEE
Number of Entries	10

6.1.61.2 Interfaces for Class basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_stringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > specified in Table 6-56, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-56 libstdcxx - Class basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > Function Interfaces

non-virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>,

allocator<wchar_t> >::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_stringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.62 Class basic_istringstream<char, char_traits<char>, allocator<char> >

6.1.62.1 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 by Table 6-57

Table 6-57 Primary vtable for basic_istringstream<char, char_traits<char>, allocator<char> >

Base Offset	0
Virtual Base Offset	48
RTTI	typeinfo for basic_istringstream<char, char_traits<char>, allocator<char> >
vfunc[0]:	basic_istringstream<char, char_traits<char>, allocator<char> >::~basic_istringstream()
vfunc[1]:	basic_istringstream<char, char_traits<char>, allocator<char> >::~basic_istringstream()

Table 6-58 Secondary vtable for basic_istringstream<char, char_traits<char>, allocator<char> >

Base Offset	-48
Virtual Base Offset	-48
RTTI	typeinfo for basic_istringstream<char, char_traits<char>, allocator<char> >
vfunc[0]:	virtual thunk to basic_istringstream<char, char_traits<char>, allocator<char> >::~basic_istringstream()
vfunc[1]:	virtual thunk to basic_istringstream<char, char_traits<char>, allocator<char> >::~basic_istringstream()

The VTT for the std::basic_istringstream<char, std::char_traits<char>, std::allocator<char> > class is described by Table 6-59

Table 6-59 VTT for basic_istringstream<char, char_traits<char>, allocator<char> >

VTT Name	_ZTTSt19basic_istringstreamIcSt11char_traitsIcESaIcEE
Number of Entries	4

6.1.62.2 Interfaces for Class basic_istringstream<char, char_traits<char>, allocator<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_istringstream<char, std::char_traits<char>, std::allocator<char> > specified in Table 6-60, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-60 libstdcxx - Class basic_istringstream<char, char_traits<char>, allocator<char> > Function Interfaces

virtual thunk to basic_istringstream<char, char_traits<char>, allocator<char> >::~basic_istringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_istringstream<char, char_traits<char>, allocator<char> >::~basic_istringstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.63 Class basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

6.1.63.1 Class data for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

The virtual table for the std::basic_istringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > class is described by Table 6-61

Table 6-61 Primary vtable for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

Base Offset	0
Virtual Base Offset	48
RTTI	typeinfo for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >
vfunc[0]:	basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_istringstream()
vfunc[1]:	basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_istringstream()

Table 6-62 Secondary vtable for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

Base Offset	-48
Virtual Base Offset	-48
RTTI	typeinfo for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >
vfunc[0]:	virtual thunk to basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_istringstream()
vfunc[1]:	virtual thunk to basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_istringstream()

The VTT for the std::basic_istringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > class is described by Table 6-63

Table 6-63 VTT for basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

VTT Name	_ZTTSt19basic_istringstreamIwSt11char_traitsIwESaIwEE
Number of Entries	4

6.1.63.2 Interfaces for Class basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_istringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > specified in Table 6-64, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-64 libstdcxx - Class basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > Function Interfaces

virtual thunk to basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_istringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_istringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_istringstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.64 Class basic_ostringstream<char, char_traits<char>, allocator<char> >

6.1.64.1 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 by Table 6-65

Table 6-65 Primary vtable for basic_ostringstream<char, char_traits<char>, allocator<char> >

Base Offset	0
Virtual Base Offset	44
RTTI	typeinfo for basic_ostringstream<char, char_traits<char>, allocator<char> >
vfunc[0]:	basic_ostringstream<char, char_traits<char>, allocator<char> >::~basic_ostringstream()
vfunc[1]:	basic_ostringstream<char, char_traits<char>, allocator<char> >::~basic_ostringstream()

Table 6-66 Secondary vtable for basic_ostringstream<char, char_traits<char>, allocator<char> >

Base Offset	-44
Virtual Base Offset	-44
RTTI	typeinfo for basic_ostringstream<char, char_traits<char>, allocator<char> >
vfunc[0]:	virtual thunk to basic_ostringstream<char, char_traits<char>, allocator<char> >::~basic_ostringstream()
vfunc[1]:	virtual thunk to basic_ostringstream<char, char_traits<char>, allocator<char> >::~basic_ostringstream()

The VTT for the std::basic_ostringstream<char, std::char_traits<char>, std::allocator<char> > class is described by Table 6-67

Table 6-67 VTT for basic_ostringstream<char, char_traits<char>, allocator<char> >

VTT Name	_ZTTSt19basic_ostringstreamIcSt11char_traitsIcESaIcEE
Number of Entries	4

6.1.64.2 Interfaces for Class basic_ostringstream<char, char_traits<char>, allocator<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ostringstream<char, std::char_traits<char>, std::allocator<char> > specified in Table 6-68, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-68 libstdcxx - Class basic_ostringstream<char, char_traits<char>, allocator<char> > Function Interfaces

virtual thunk to basic_ostringstream<char, char_traits<char>, allocator<char> >::~basic_ostringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ostringstream<char, char_traits<char>, allocator<char> >::~basic_ostringstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.65 Class basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

6.1.65.1 Class data for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

The virtual table for the std::basic_ostringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > class is described by Table 6-69

Table 6-69 Primary vtable for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

Base Offset	0
Virtual Base Offset	44
RTTI	typeinfo for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >
vfunc[0]:	basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_ostringstream()
vfunc[1]:	basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_ostringstream()

Table 6-70 Secondary vtable for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

Base Offset	-44
Virtual Base Offset	-44
RTTI	typeinfo for basic_ostringstream<wchar_t, char_traits<wchar_t>,

	allocator<wchar_t> >
vfunc[0]:	virtual thunk to basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_ostringstream()
vfunc[1]:	virtual thunk to basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_ostringstream()

The VTT for the std::basic_ostringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > class is described by Table 6-71

Table 6-71 VTT for basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

VTT Name	_ZTTSt19basic_ostringstreamIwSt11char_traitsIwESaIwEE
Number of Entries	4

6.1.65.2 Interfaces for Class basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ostringstream<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > specified in Table 6-72, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-72 libstdcxx - Class basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > Function Interfaces

virtual thunk to basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_ostringstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ostringstream<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_ostringstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.66 Class basic_stringbuf<char, char_traits<char>, allocator<char> >

6.1.66.1 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 6-73

Table 6-73 Primary vtable for basic_stringbuf<char, char_traits<char>, allocator<char> >

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for basic_stringbuf<char,

	char_traits<char>, allocator<char> >
vfunc[0]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::~basic_stringbuf()
vfunc[1]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::~basic_stringbuf()
vfunc[2]:	basic_streambuf<char, char_traits<char> >::imbue(locale const&)
vfunc[3]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::setbuf(char*, int)
vfunc[4]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)
vfunc[5]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_streambuf<char, char_traits<char> >::sync()
vfunc[7]:	basic_streambuf<char, char_traits<char> >::showmanyc()
vfunc[8]:	basic_streambuf<char, char_traits<char> >::xsgetn(char*, int)
vfunc[9]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::underflow()
vfunc[10]:	basic_streambuf<char, char_traits<char> >::uflow()
vfunc[11]:	basic_stringbuf<char, char_traits<char>, allocator<char> >::pbackfail(int)
vfunc[12]:	basic_streambuf<char, char_traits<char> >::xsputn(char const*, int)
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 6-74

Table 6-74 typeinfo for basic_stringbuf<char, char_traits<char>, allocator<char> >

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

6.1.66.2 Interfaces for Class basic_stringbuf<char, char_traits<char>, allocator<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_stringbuf<char, std::char_traits<char>, std::allocator<char> > specified in Table 6-75, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-75 libstdcxx - Class basic_stringbuf<char, char_traits<char>, allocator<char> > Function Interfaces

basic_stringbuf<char, char_traits<char>, allocator<char> >::setbuf(char*, int)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char> >::_M_sync(char*, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<char, char_traits<char>, allocator<char> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]

6.1.67 Class basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

6.1.67.1 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 6-76

Table 6-76 Primary vtable for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >
vfunc[0]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::~basic_stringbuf()
vfunc[1]:	basic_stringbuf<wchar_t, char_traits<wchar_t>,

	allocator<wchar_t> >::~basic_stringbuf()
vfunc[2]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::imbue(locale const&)
vfunc[3]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::setbuf(wchar_t*, int)
vfunc[4]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)
vfunc[5]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::sync()
vfunc[7]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::showmanyc()
vfunc[8]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::xsgetn(wchar_t*, int)
vfunc[9]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::underflow()
vfunc[10]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::uflow()
vfunc[11]:	basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::pbackfail(unsigned int)
vfunc[12]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::xsputn(wchar_t const*, int)
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 6-77

Table 6-77 typeinfo for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

6.1.67.2 Interfaces for Class basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_stringbuf<wchar_t, std::char_traits<wchar_t>, std::allocator<wchar_t> > specified in Table 6-78, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-78 libstdcxx - Class basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> > Function Interfaces

basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::setbuf(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::_M_sync(wchar_t*, unsigned int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_stringbuf<wchar_t, char_traits<wchar_t>, allocator<wchar_t> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]

6.1.68 Class basic_iostream<char, char_traits<char> >

6.1.68.1 Class data for basic_iostream<char, char_traits<char> >

The virtual table for the std::basic_iostream<char, std::char_traits<char> > class is described by Table 6-79

Table 6-79 Primary vtable for basic_iostream<char, char_traits<char> >

Base Offset	0
Virtual Base Offset	12
RTTI	typeinfo for basic_iostream<char, char_traits<char> >
vfunc[0]:	basic_iostream<char, char_traits<char> >::~basic_iostream()
vfunc[1]:	basic_iostream<char, char_traits<char> >::~basic_iostream()

Table 6-80 Secondary vtable for basic_iostream<char, char_traits<char> >

Base Offset	-8

Virtual Base Offset	4
RTTI	typeinfo for basic_iostream<char, char_traits<char> >
vfunc[0]:	non-virtual thunk to basic_iostream<char, char_traits<char> >::~basic_iostream()
vfunc[1]:	non-virtual thunk to basic_iostream<char, char_traits<char> >::~basic_iostream()

Table 6-81 Secondary vtable for basic_iostream<char, char_traits<char> >

Base Offset	-12
Virtual Base Offset	-12
RTTI	typeinfo for basic_iostream<char, char_traits<char> >
vfunc[0]:	virtual thunk to basic_iostream<char, char_traits<char> >::~basic_iostream()
vfunc[1]:	virtual thunk to basic_iostream<char, char_traits<char> >::~basic_iostream()

The VTT for the std::basic_iostream<char, std::char_traits<char> > class is described by Table 6-82

Table 6-82 VTT for basic_iostream<char, char_traits<char> >

VTT Name	_ZTTSd
Number of Entries	7

6.1.68.2 Interfaces for Class basic_iostream<char, char_traits<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_iostream<char, std::char_traits<char> > specified in Table 6-83, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-83 libstdcxx - Class basic_iostream<char, char_traits<char> > Function Interfaces

non-virtual thunk to basic_iostream<char, char_traits<char> >::~basic_iostream()(GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to basic_iostream<char, char_traits<char> >::~basic_iostream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_iostream<char, char_traits<char>

>::~basic_iostream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_iostream<char, char_traits<char> >::~basic_iostream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.69 Class basic_iostream<wchar_t, char_traits<wchar_t> >

6.1.69.1 Class data for basic_iostream<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_iostream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-84

Table 6-84 Primary vtable for basic_iostream<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	12
RTTI	typeinfo for basic_iostream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_iostream<wchar_t, char_traits<wchar_t> >::~basic_iostream()
vfunc[1]:	basic_iostream<wchar_t, char_traits<wchar_t> >::~basic_iostream()

Table 6-85 Secondary vtable for basic_iostream<wchar_t, char_traits<wchar_t> >

Base Offset	-8
Virtual Base Offset	4
RTTI	typeinfo for basic_iostream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	non-virtual thunk to basic_iostream<wchar_t, char_traits<wchar_t> >::~basic_iostream()
vfunc[1]:	non-virtual thunk to basic_iostream<wchar_t, char_traits<wchar_t> >::~basic_iostream()

Table 6-86 Secondary vtable for basic_iostream<wchar_t, char_traits<wchar_t> >

Base Offset	-12
Virtual Base Offset	-12
RTTI	typeinfo for basic_iostream<wchar_t, char_traits<wchar_t> >

vfunc[0]:	virtual thunk to basic_iostream<wchar_t, char_traits<wchar_t> >::~basic_iostream()
vfunc[1]:	virtual thunk to basic_iostream<wchar_t, char_traits<wchar_t> >::~basic_iostream()

The VTT for the std::basic_iostream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-87

Table 6-87 VTT for basic_iostream<wchar_t, char_traits<wchar_t> >

VTT Name	_ZTTSt14basic_iostreamIwSt11char_traitsIwEE
Number of Entries	7

6.1.69.2 Interfaces for Class basic_iostream<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_iostream<wchar_t, std::char_traits<wchar_t> > specified in Table 6-88, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-88 libstdcxx - Class basic_iostream<wchar_t, char_traits<wchar_t> > Function Interfaces

non-virtual thunk to basic_iostream<wchar_t, char_traits<wchar_t> >::~basic_iostream()(GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to basic_iostream<wchar_t, char_traits<wchar_t> >::~basic_iostream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_iostream<wchar_t, char_traits<wchar_t> >::~basic_iostream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_iostream<wchar_t, char_traits<wchar_t> >::~basic_iostream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.70 Class basic_istream<char, char_traits<char> >

6.1.70.1 Class data for basic_istream<char, char_traits<char> >

The virtual table for the std::basic_istream<char, std::char_traits<char> > class is described by Table 6-89

Table 6-89 Primary vtable for basic_istream<char, char_traits<char> >

Base Offset	0
Virtual Base Offset	8
RTTI	typeinfo for basic_istream<char, char_traits<char> >

vfunc[0]:	basic_istream<char, char_traits<char> >::~basic_istream()
vfunc[1]:	basic_istream<char, char_traits<char> >::~basic_istream()

Table 6-90 Secondary vtable for basic_istream<char, char_traits<char> >

Base Offset	-8
Virtual Base Offset	-8
RTTI	typeinfo for basic_istream<char, char_traits<char> >
vfunc[0]:	virtual thunk to basic_istream<char, char_traits<char> >::~basic_istream()
vfunc[1]:	virtual thunk to basic_istream<char, char_traits<char> >::~basic_istream()

The VTT for the std::basic_istream<char, std::char_traits<char> > class is described by Table 6-91

Table 6-91 VTT for basic_istream<char, char_traits<char> >

VTT Name	_ZTTSi
Number of Entries	2

6.1.70.2 Interfaces for Class basic_istream<char, char_traits<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_istream<char, std::char_traits<char> > specified in Table 6-92, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-92 libstdcxx - Class basic_istream<char, char_traits<char> > Function Interfaces

basic_istream<char, char_traits<char> >::get(char*, int)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >::get(char*, int, char)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >::read(char*, int)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >::seekg(long long, _Ios_Seekdir)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >::ignore(int)(GLIBCXX_3.4.5) [ISOCXX]
basic_istream<char, char_traits<char> >::ignore(int, int)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >::getline(char*, int)(GLIBCXX_3.4)

[ISOCXX]
basic_istream<char, char_traits<char> >::getline(char*, int, char)(GLIBCXX_3.4) [ISOCXX]
basic_istream<char, char_traits<char> >::readsome(char*, int)(GLIBCXX_3.4) [ISOCXX]
virtual thunk to basic_istream<char, char_traits<char> >::~basic_istream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_istream<char, char_traits<char> >::~basic_istream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.71 Class basic_istream<wchar_t, char_traits<wchar_t> >

6.1.71.1 Class data for basic_istream<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_istream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-93

Table 6-93 Primary vtable for basic_istream<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	8
RTTI	typeinfo for basic_istream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()
vfunc[1]:	basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()

Table 6-94 Secondary vtable for basic_istream<wchar_t, char_traits<wchar_t> >

Base Offset	-8
Virtual Base Offset	-8
RTTI	typeinfo for basic_istream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	virtual thunk to basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()
vfunc[1]:	virtual thunk to basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()

The VTT for the std::basic_istream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-95

Table 6-95 VTT for basic_istream<wchar_t, char_traits<wchar_t> >

VTT Name	_ZTTSt13basic_istreamIwSt11char_traitsIwEE
Number of Entries	2

6.1.71.2 Interfaces for Class basic_istream<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_istream<wchar_t, std::char_traits<wchar_t> > specified in Table 6-96, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-96 libstdcxx - Class basic_istream<wchar_t, char_traits<wchar_t> > Function Interfaces

basic_istream<wchar_t, char_traits<wchar_t> >::get(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t> >::get(wchar_t*, int, wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t> >::read(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t> >::ignore(int)(GLIBCXX_3.4.5) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t> >::ignore(int, unsigned int)(GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t> >::getline(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t> >::getline(wchar_t*, int, wchar_t)(GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t> >::readsome(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]
virtual thunk to basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_istream<wchar_t, char_traits<wchar_t> >::~basic_istream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.72 Class istreambuf_iterator<wchar_t, char_traits<wchar_t> >

6.1.72.1 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 generic specification.

6.1.73 Class istreambuf_iterator<char, char_traits<char> >

6.1.73.1 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 generic specification.

6.1.74 Class basic_ostream<char, char_traits<char> >

6.1.74.1 Class data for basic_ostream<char, char_traits<char> >

The virtual table for the std::basic_ostream<char, std::char_traits<char> > class is described by Table 6-97

Table 6-97 Primary vtable for basic_ostream<char, char_traits<char> >

Base Offset	0
Virtual Base Offset	4
RTTI	typeinfo for basic_ostream<char, char_traits<char> >
vfunc[0]:	basic_ostream<char, char_traits<char> >::~basic_ostream()
vfunc[1]:	basic_ostream<char, char_traits<char> >::~basic_ostream()

Table 6-98 Secondary vtable for basic_ostream<char, char_traits<char> >

Base Offset	-4
Virtual Base Offset	-4
RTTI	typeinfo for basic_ostream<char, char_traits<char> >
vfunc[0]:	virtual thunk to basic_ostream<char, char_traits<char> >::~basic_ostream()
vfunc[1]:	virtual thunk to basic_ostream<char, char_traits<char> >::~basic_ostream()

The VTT for the std::basic_ostream<char, std::char_traits<char> > class is described by Table 6-99

Table 6-99 VTT for basic_ostream<char, char_traits<char> >

VTT Name	_ZTTSo
Number of Entries	2

6.1.74.2 Interfaces for Class basic_ostream<char, char_traits<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ostream<char, std::char_traits<char> > specified in Table 6-100, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-100 libstdcxx - Class basic_ostream<char, char_traits<char> > Function Interfaces

basic_ostream<char, char_traits<char> >::seekp(long long, _Ios_Seekdir)(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::write(char const*, int)(GLIBCXX_3.4) [ISOCXX]
basic_ostream<char, char_traits<char> >::_M_write(char const*, int)(GLIBCXX_3.4) [ISOCXX]
virtual thunk to basic_ostream<char, char_traits<char> >::~basic_ostream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ostream<char, char_traits<char> >::~basic_ostream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.75 Class basic_ostream<wchar_t, char_traits<wchar_t> >

6.1.75.1 Class data for basic_ostream<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_ostream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-101

Table 6-101 Primary vtable for basic_ostream<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	4
RTTI	typeinfo for basic_ostream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_ostream<wchar_t, char_traits<wchar_t> >::~basic_ostream()
vfunc[1]:	basic_ostream<wchar_t, char_traits<wchar_t> >::~basic_ostream()

Table 6-102 Secondary vtable for basic_ostream<wchar_t, char_traits<wchar_t> >

Base Offset	-4
Virtual Base Offset	-4
RTTI	typeinfo for basic_ostream<wchar_t, char_traits<wchar_t> >

vfunc[0]:	virtual thunk to basic_ostream<wchar_t, char_traits<wchar_t> >::~basic_ostream()
vfunc[1]:	virtual thunk to basic_ostream<wchar_t, char_traits<wchar_t> >::~basic_ostream()

The VTT for the std::basic_ostream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-103

Table 6-103 VTT for basic_ostream<wchar_t, char_traits<wchar_t> >

VTT Name	_ZTTSt13basic_ostreamIwSt11char_traitsIwEE
Number of Entries	2

6.1.75.2 Interfaces for Class basic_ostream<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ostream<wchar_t, std::char_traits<wchar_t> > specified in Table 6-104, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-104 libstdcxx - Class basic_ostream<wchar_t, char_traits<wchar_t> > Function Interfaces

basic_ostream<wchar_t, char_traits<wchar_t> >::write(wchar_t const*, int)(GLIBCXX_3.4) [ISOCXX]
virtual thunk to basic_ostream<wchar_t, char_traits<wchar_t> >::~basic_ostream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ostream<wchar_t, char_traits<wchar_t> >::~basic_ostream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.76 Class basic_fstream<char, char_traits<char> >

6.1.76.1 Class data for basic_fstream<char, char_traits<char> >

The virtual table for the std::basic_fstream<char, std::char_traits<char> > class is described by Table 6-105

Table 6-105 Primary vtable for basic_fstream<char, char_traits<char> >

Base Offset	0
Virtual Base Offset	148
RTTI	typeinfo for basic_fstream<char, char_traits<char> >
vfunc[0]:	basic_fstream<char, char_traits<char> >::~basic_fstream()

vfunc[1]:	basic_fstream<char, char_traits<char> >::~basic_fstream()

Table 6-106 Secondary vtable for basic_fstream<char, char_traits<char> >

Base Offset	-8
Virtual Base Offset	140
RTTI	typeinfo for basic_fstream<char, char_traits<char> >
vfunc[0]:	non-virtual thunk to basic_fstream<char, char_traits<char> >::~basic_fstream()
vfunc[1]:	non-virtual thunk to basic_fstream<char, char_traits<char> >::~basic_fstream()

Table 6-107 Secondary vtable for basic_fstream<char, char_traits<char> >

Base Offset	-148
Virtual Base Offset	-148
RTTI	typeinfo for basic_fstream<char, char_traits<char> >
vfunc[0]:	virtual thunk to basic_fstream<char, char_traits<char> >::~basic_fstream()
vfunc[1]:	virtual thunk to basic_fstream<char, char_traits<char> >::~basic_fstream()

The VTT for the std::basic_fstream<char, std::char_traits<char> > class is described by Table 6-108

Table 6-108 VTT for basic_fstream<char, char_traits<char> >

VTT Name	_ZTTSt13basic_fstreamIcSt11char_traitsIcEE
Number of Entries	10

6.1.76.2 Interfaces for Class basic_fstream<char, char_traits<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_fstream<char, std::char_traits<char> > specified in Table 6-109, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-109 libstdcxx - Class basic_fstream<char, char_traits<char> > Function Interfaces

non-virtual thunk to basic_fstream<char, char_traits<char> >::~basic_fstream()(GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to basic_fstream<char, char_traits<char>

>::~basic_fstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_fstream<char, char_traits<char> >::~basic_fstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_fstream<char, char_traits<char> >::~basic_fstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.77 Class basic_fstream<wchar_t, char_traits<wchar_t> >

6.1.77.1 Class data for basic_fstream<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_fstream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-110

Table 6-110 Primary vtable for basic_fstream<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	152
RTTI	typeinfo for basic_fstream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_fstream<wchar_t, char_traits<wchar_t> >::~basic_fstream()
vfunc[1]:	basic_fstream<wchar_t, char_traits<wchar_t> >::~basic_fstream()

Table 6-111 Secondary vtable for basic_fstream<wchar_t, char_traits<wchar_t> >

Base Offset	-8
Virtual Base Offset	144
RTTI	typeinfo for basic_fstream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	non-virtual thunk to basic_fstream<wchar_t, char_traits<wchar_t> >::~basic_fstream()
vfunc[1]:	non-virtual thunk to basic_fstream<wchar_t, char_traits<wchar_t> >::~basic_fstream()

Table 6-112 Secondary vtable for basic_fstream<wchar_t, char_traits<wchar_t> >

Base Offset	-152
Virtual Base Offset	-152

RTTI	typeinfo for basic_fstream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	virtual thunk to basic_fstream<wchar_t, char_traits<wchar_t> >::~basic_fstream()
vfunc[1]:	virtual thunk to basic_fstream<wchar_t, char_traits<wchar_t> >::~basic_fstream()

The VTT for the std::basic_fstream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-113

Table 6-113 VTT for basic_fstream<wchar_t, char_traits<wchar_t> >

VTT Name	_ZTTSt13basic_fstreamIwSt11char_traitsIwEE
Number of Entries	10

6.1.77.2 Interfaces for Class basic_fstream<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_fstream<wchar_t, std::char_traits<wchar_t> > specified in Table 6-114, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-114 libstdcxx - Class basic_fstream<wchar_t, char_traits<wchar_t> > Function Interfaces

non-virtual thunk to basic_fstream<wchar_t, char_traits<wchar_t> >::~basic_fstream()(GLIBCXX_3.4) [CXXABI-1.86]
non-virtual thunk to basic_fstream<wchar_t, char_traits<wchar_t> >::~basic_fstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_fstream<wchar_t, char_traits<wchar_t> >::~basic_fstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_fstream<wchar_t, char_traits<wchar_t> >::~basic_fstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.78 Class basic_ifstream<char, char_traits<char> >

6.1.78.1 Class data for basic_ifstream<char, char_traits<char> >

The virtual table for the std::basic_ifstream<char, std::char_traits<char> > class is described by Table 6-115

Table 6-115 Primary vtable for basic_ifstream<char, char_traits<char> >

Base Offset	0
Virtual Base Offset	144

RTTI	typeinfo for basic_ifstream<char, char_traits<char> >
vfunc[0]:	basic_ifstream<char, char_traits<char> >::~basic_ifstream()
vfunc[1]:	basic_ifstream<char, char_traits<char> >::~basic_ifstream()

Table 6-116 Secondary vtable for basic_ifstream<char, char_traits<char> >

Base Offset	-144
Virtual Base Offset	-144
RTTI	typeinfo for basic_ifstream<char, char_traits<char> >
vfunc[0]:	virtual thunk to basic_ifstream<char, char_traits<char> >::~basic_ifstream()
vfunc[1]:	virtual thunk to basic_ifstream<char, char_traits<char> >::~basic_ifstream()

The VTT for the std::basic_ifstream<char, std::char_traits<char> > class is described by Table 6-117

Table 6-117 VTT for basic_ifstream<char, char_traits<char> >

VTT Name	_ZTTSt14basic_ifstreamIcSt11char_traitsIcEE
Number of Entries	4

6.1.78.2 Interfaces for Class basic_ifstream<char, char_traits<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ifstream<char, std::char_traits<char> > specified in Table 6-118, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-118 libstdcxx - Class basic_ifstream<char, char_traits<char> > Function Interfaces

virtual thunk to basic_ifstream<char, char_traits<char> >::~basic_ifstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ifstream<char, char_traits<char> >::~basic_ifstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.79 Class basic_ifstream<wchar_t, char_traits<wchar_t> >

6.1.79.1 Class data for basic_ifstream<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_ifstream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-119

Table 6-119 Primary vtable for basic_ifstream<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	148
RTTI	typeinfo for basic_ifstream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_ifstream<wchar_t, char_traits<wchar_t> >::~basic_ifstream()
vfunc[1]:	basic_ifstream<wchar_t, char_traits<wchar_t> >::~basic_ifstream()

Table 6-120 Secondary vtable for basic_ifstream<wchar_t, char_traits<wchar_t> >

Base Offset	-148
Virtual Base Offset	-148
RTTI	typeinfo for basic_ifstream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	virtual thunk to basic_ifstream<wchar_t, char_traits<wchar_t> >::~basic_ifstream()
vfunc[1]:	virtual thunk to basic_ifstream<wchar_t, char_traits<wchar_t> >::~basic_ifstream()

The VTT for the std::basic_ifstream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-121

Table 6-121 VTT for basic_ifstream<wchar_t, char_traits<wchar_t> >

VTT Name	_ZTTSt14basic_ifstreamIwSt11char_traitsIwEE
Number of Entries	4

6.1.79.2 Interfaces for Class basic_ifstream<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ifstream<wchar_t, std::char_traits<wchar_t> > specified in Table 6-122, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-122 libstdcxx - Class basic_ifstream<wchar_t, char_traits<wchar_t> > Function Interfaces

virtual thunk to basic_ifstream<wchar_t, char_traits<wchar_t> >::~basic_ifstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ifstream<wchar_t, char_traits<wchar_t> >::~basic_ifstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.80 Class basic_ofstream<char, char_traits<char> >

6.1.80.1 Class data for basic_ofstream<char, char_traits<char> >

The virtual table for the std::basic_ofstream<char, std::char_traits<char> > class is described by Table 6-123

Table 6-123 Primary vtable for basic_ofstream<char, char_traits<char> >

Base Offset	0
Virtual Base Offset	140
RTTI	typeinfo for basic_ofstream<char, char_traits<char> >
vfunc[0]:	basic_ofstream<char, char_traits<char> >::~basic_ofstream()
vfunc[1]:	basic_ofstream<char, char_traits<char> >::~basic_ofstream()

Table 6-124 Secondary vtable for basic_ofstream<char, char_traits<char> >

Base Offset	-140
Virtual Base Offset	-140
RTTI	typeinfo for basic_ofstream<char, char_traits<char> >
vfunc[0]:	virtual thunk to basic_ofstream<char, char_traits<char> >::~basic_ofstream()
vfunc[1]:	virtual thunk to basic_ofstream<char, char_traits<char> >::~basic_ofstream()

The VTT for the std::basic_ofstream<char, std::char_traits<char> > class is described by Table 6-125

Table 6-125 VTT for basic_ofstream<char, char_traits<char> >

VTT Name	_ZTTSt14basic_ofstreamIcSt11char_traitsIcEE
Number of Entries	4

6.1.80.2 Interfaces for Class basic_ofstream<char, char_traits<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ofstream<char, std::char_traits<char> > specified in Table 6-126, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-126 libstdcxx - Class basic_ofstream<char, char_traits<char> > Function Interfaces

virtual thunk to basic_ofstream<char, char_traits<char> >::~basic_ofstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ofstream<char, char_traits<char> >::~basic_ofstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.81 Class basic_ofstream<wchar_t, char_traits<wchar_t> >

6.1.81.1 Class data for basic_ofstream<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_ofstream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-127

Table 6-127 Primary vtable for basic_ofstream<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	144
RTTI	typeinfo for basic_ofstream<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()
vfunc[1]:	basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()

Table 6-128 Secondary vtable for basic_ofstream<wchar_t, char_traits<wchar_t> >

Base Offset	-144
Virtual Base Offset	-144
RTTI	typeinfo for basic_ofstream<wchar_t, char_traits<wchar_t> >

vfunc[0]:	virtual thunk to basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()
vfunc[1]:	virtual thunk to basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()

The VTT for the std::basic_ofstream<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-129

Table 6-129 VTT for basic_ofstream<wchar_t, char_traits<wchar_t> >

VTT Name	_ZTTSt14basic_ofstreamIwSt11char_traitsIwEE
Number of Entries	4

6.1.81.2 Interfaces for Class basic_ofstream<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_ofstream<wchar_t, std::char_traits<wchar_t> > specified in Table 6-130, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-130 libstdcxx - Class basic_ofstream<wchar_t, char_traits<wchar_t> > Function Interfaces

virtual thunk to basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()(GLIBCXX_3.4) [CXXABI-1.86]
virtual thunk to basic_ofstream<wchar_t, char_traits<wchar_t> >::~basic_ofstream()(GLIBCXX_3.4) [CXXABI-1.86]

6.1.82 Class basic_streambuf<char, char_traits<char> >

6.1.82.1 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 6-131

Table 6-131 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]:	basic_streambuf<char, char_traits<char> >::setbuf(char*, int)
vfunc[4]:	basic_streambuf<char, char_traits<char> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)
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]:	basic_streambuf<char, char_traits<char> >::xsgetn(char*, int)
vfunc[9]:	basic_streambuf<char, char_traits<char> >::underflow()
vfunc[10]:	basic_streambuf<char, char_traits<char> >::uflow()
vfunc[11]:	basic_streambuf<char, char_traits<char> >::pbackfail(int)
vfunc[12]:	basic_streambuf<char, char_traits<char> >::xsputn(char const*, int)
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 6-132

Table 6-132 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> >

6.1.82.2 Interfaces for Class basic_streambuf<char, char_traits<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_streambuf<char, std::char_traits<char> > specified

in Table 6-133, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-133 libstdcxx - Class basic_streambuf<char, char_traits<char> > Function Interfaces

basic_streambuf<char, char_traits<char> >::pubseekoff(long long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::sgetn(char*, int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::sputn(char const*, int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::setbuf(char*, int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::xsgetn(char*, int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::xsputn(char const*, int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<char, char_traits<char> >::pubsetbuf(char*, int)(GLIBCXX_3.4) [ISOCXX]

6.1.83 Class basic_streambuf<wchar_t, char_traits<wchar_t> >

6.1.83.1 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 6-134

Table 6-134 Primary vtable for basic_streambuf<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for basic_streambuf<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::~basic_streambuf()
vfunc[1]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::~basic_streambuf()
vfunc[2]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::imbue(locale const&)

vfunc[3]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::setbuf(wchar_t*, int)
vfunc[4]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)
vfunc[5]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::sync()
vfunc[7]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::showmanyc()
vfunc[8]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::xsgetn(wchar_t*, int)
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]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::xsputn(wchar_t const*, int)
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 6-135

Table 6-135 typeinfo for basic_streambuf<wchar_t, char_traits<wchar_t> >

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for basic_streambuf<wchar_t, char_traits<wchar_t> >

6.1.83.2 Interfaces for Class basic_streambuf<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_streambuf<wchar_t, std::char_traits<wchar_t> >

specified in Table 6-136, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-136 libstdcxx - Class basic_streambuf<wchar_t, char_traits<wchar_t> > Function Interfaces

basic_streambuf<wchar_t, char_traits<wchar_t> >::pubseekoff(long long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::sgetn(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::sputn(wchar_t const*, int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::setbuf(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::xsgetn(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::xsputn(wchar_t const*, int)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_streambuf<wchar_t, char_traits<wchar_t> >::pubsetbuf(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]

6.1.84 Class basic_filebuf<char, char_traits<char> >

6.1.84.1 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 6-137

Table 6-137 Primary vtable for basic_filebuf<char, char_traits<char> >

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for basic_filebuf<char, char_traits<char> >
vfunc[0]:	basic_filebuf<char, char_traits<char> >::~basic_filebuf()
vfunc[1]:	basic_filebuf<char, char_traits<char> >::~basic_filebuf()
vfunc[2]:	basic_filebuf<char, char_traits<char> >::imbue(locale const&)
vfunc[3]:	basic_filebuf<char, char_traits<char> >::setbuf(char*, int)
vfunc[4]:	basic_filebuf<char, char_traits<char> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)

vfunc[5]:	basic_filebuf<char, char_traits<char> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_filebuf<char, char_traits<char> >::sync()
vfunc[7]:	basic_filebuf<char, char_traits<char> >::showmanyc()
vfunc[8]:	basic_filebuf<char, char_traits<char> >::xsgetn(char*, int)
vfunc[9]:	basic_filebuf<char, char_traits<char> >::underflow()
vfunc[10]:	basic_streambuf<char, char_traits<char> >::uflow()
vfunc[11]:	basic_filebuf<char, char_traits<char> >::pbackfail(int)
vfunc[12]:	basic_filebuf<char, char_traits<char> >::xsputn(char const*, int)
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 6-138

Table 6-138 typeinfo for basic_filebuf<char, char_traits<char> >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for basic_filebuf<char, char_traits<char> >

6.1.84.2 Interfaces for Class basic_filebuf<char, char_traits<char> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_filebuf<char, std::char_traits<char> > specified in Table 6-139, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-139 libstdcxx - Class basic_filebuf<char, char_traits<char> > Function Interfaces

basic_filebuf<char, char_traits<char> >::_M_set_buffer(int)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<char, char_traits<char> >::_M_convert_to_external(char*, int)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<char, char_traits<char> >::setbuf(char*, int)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<char, char_traits<char> >::xsgetn(char*, int)(GLIBCXX_3.4) [ISOCXX]

basic_filebuf<char, char_traits<char> >::xsputn(char const*, int)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<char, char_traits<char> >::_M_seek(long long, _Ios_Seekdir, __mbstate_t)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<char, char_traits<char> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]

6.1.85 Class basic_filebuf<wchar_t, char_traits<wchar_t> >

6.1.85.1 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 6-140

Table 6-140 Primary vtable for basic_filebuf<wchar_t, char_traits<wchar_t> >

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for basic_filebuf<wchar_t, char_traits<wchar_t> >
vfunc[0]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::~basic_filebuf()
vfunc[1]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::~basic_filebuf()
vfunc[2]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::imbue(locale const&)
vfunc[3]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::setbuf(wchar_t*, int)
vfunc[4]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)
vfunc[5]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::seekpos(fpos<__mbstate_t>, _Ios_Openmode)
vfunc[6]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::sync()
vfunc[7]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::showmanyc()
vfunc[8]:	basic_filebuf<wchar_t, char_traits<wchar_t>

	>::xsgetn(wchar_t*, int)
vfunc[9]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::underflow()
vfunc[10]:	basic_streambuf<wchar_t, char_traits<wchar_t> >::uflow()
vfunc[11]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::pbackfail(unsigned int)
vfunc[12]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::xsputn(wchar_t const*, int)
vfunc[13]:	basic_filebuf<wchar_t, char_traits<wchar_t> >::overflow(unsigned int)

The Run Time Type Information for the std::basic_filebuf<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-141

Table 6-141 typeinfo for basic_filebuf<wchar_t, char_traits<wchar_t> >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for basic_filebuf<wchar_t, char_traits<wchar_t> >

6.1.85.2 Interfaces for Class basic_filebuf<wchar_t, char_traits<wchar_t> >

An LSB conforming implementation shall provide the architecture specific methods for Class std::basic_filebuf<wchar_t, std::char_traits<wchar_t> > specified in Table 6-142, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-142 libstdcxx - Class basic_filebuf<wchar_t, char_traits<wchar_t> > Function Interfaces

basic_filebuf<wchar_t, char_traits<wchar_t> >::_M_set_buffer(int)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::_M_convert_to_external(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::setbuf(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::xsgetn(wchar_t*, int)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::xsputn(wchar_t const*, int)(GLIBCXX_3.4) [ISOCXX]
basic_filebuf<wchar_t, char_traits<wchar_t> >::_M_seek(long long, _Ios_Seekdir, __mbstate_t)(GLIBCXX_3.4) [ISOCXX]

basic_filebuf<wchar_t, char_traits<wchar_t> >::seekoff(long long, _Ios_Seekdir, _Ios_Openmode)(GLIBCXX_3.4) [ISOCXX]
basic_istream<wchar_t, char_traits<wchar_t> >::seekg(long long, _Ios_Seekdir)(GLIBCXX_3.4) [ISOCXX]
basic_ostream<wchar_t, char_traits<wchar_t> >::seekp(long long, _Ios_Seekdir)(GLIBCXX_3.4) [ISOCXX]
basic_ostream<wchar_t, char_traits<wchar_t> >::_M_write(wchar_t const*, int)(GLIBCXX_3.4) [ISOCXX]

6.1.86 Class ios_base

6.1.86.1 Class data for ios_base

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

The Run Time Type Information for the std::ios_base class is described by Table 6-143

Table 6-143 typeinfo for ios_base

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for ios_base

6.1.86.2 Interfaces for Class ios_base

No external methods are defined for libstdcxx - Class std::ios_base in this part of the specification. See also the generic specification.

6.1.87 Class basic_ios<char, char_traits<char> >

6.1.87.1 Class data for basic_ios<char, char_traits<char> >

The virtual table for the std::basic_ios<char, std::char_traits<char> > class is described in the generic part of this specification.

6.1.87.2 Interfaces for Class basic_ios<char, char_traits<char> >

No external methods are defined for libstdcxx - Class std::basic_ios<char, std::char_traits<char> > in this part of the specification. See also the generic specification.

6.1.88 Class basic_ios<wchar_t, char_traits<wchar_t> >

6.1.88.1 Class data for basic_ios<wchar_t, char_traits<wchar_t> >

The virtual table for the std::basic_ios<wchar_t, std::char_traits<wchar_t> > class is described in the generic part of this specification.

The Run Time Type Information for the std::basic_ios<wchar_t, std::char_traits<wchar_t> > class is described by Table 6-144

Table 6-144 typeinfo for basic_ios<wchar_t, char_traits<wchar_t> >

Base Vtable	vtable for __cxxabiv1::__si_class_t	

	ype_info	
Name	typeinfo name for basic_ios<wchar_t, char_traits<wchar_t> >	
flags:	8	
basetype:	typeinfo for ios_base	1026

6.1.88.2 Interfaces for Class basic_ios<wchar_t, char_traits<wchar_t> >

No external methods are defined for libstdcxx - Class std::basic_ios<wchar_t, std::char_traits<wchar_t> > in this part of the specification. See also the generic specification.

6.1.89 Class ios_base::failure

6.1.89.1 Class data for ios_base::failure

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

The Run Time Type Information for the std::ios_base::failure class is described by Table 6-145

Table 6-145 typeinfo for ios_base::failure

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for ios_base::failure

6.1.89.2 Interfaces for Class ios_base::failure

No external methods are defined for libstdcxx - Class std::ios_base::failure in this part of the specification. See also the generic specification.

6.1.90 Class __timepunct<char>

6.1.90.1 Class data for __timepunct<char>

The virtual table for the std::__timepunct<char> class is described in the generic part of this specification.

The Run Time Type Information for the std::__timepunct<char> class is described by Table 6-146

Table 6-146 typeinfo for __timepunct<char>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for __timepunct<char>

6.1.90.2 Interfaces for Class __timepunct<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::__timepunct<char> specified in Table 6-147, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-147 libstdcxx - Class __timepunct<char> Function Interfaces

__timepunct<char>::_M_put(char*, unsigned int, char const*, tm const*) const(GLIBCXX_3.4) [ISOCXX]
__timepunct<char>::__timepunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
__timepunct<char>::__timepunct(__timepunct_cache<char>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
__timepunct<char>::__timepunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]
__timepunct<char>::__timepunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
__timepunct<char>::__timepunct(__timepunct_cache<char>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
__timepunct<char>::__timepunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.91 Class __timepunct<wchar_t>

6.1.91.1 Class data for __timepunct<wchar_t>

The virtual table for the std::__timepunct<wchar_t> class is described in the generic part of this specification.

The Run Time Type Information for the std::__timepunct<wchar_t> class is described by Table 6-148

Table 6-148 typeinfo for __timepunct<wchar_t>

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

6.1.91.2 Interfaces for Class __timepunct<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::__timepunct<wchar_t> specified in Table 6-149, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-149 libstdcxx - Class __timepunct<wchar_t> Function Interfaces

__timepunct<wchar_t>::_M_put(wchar_t*, unsigned int, wchar_t const*, tm const*) const(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::__timepunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::__timepunct(__timepunct_cache<wchar_t>*,

unsigned int)(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::__timepunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::__timepunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::__timepunct(__timepunct_cache<wchar_t>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
__timepunct<wchar_t>::__timepunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.92 Class messages_base

6.1.92.1 Class data for messages_base

The Run Time Type Information for the std::messages_base class is described by Table 6-150

Table 6-150 typeinfo for messages_base

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for messages_base

6.1.92.2 Interfaces for Class messages_base

No external methods are defined for libstdcxx - Class std::messages_base in this part of the specification. See also the generic specification.

6.1.93 Class messages<char>

6.1.93.1 Class data for messages<char>

The virtual table for the std::messages<char> class is described in the generic part of this specification.

6.1.93.2 Interfaces for Class messages<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::messages<char> specified in Table 6-151, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-151 libstdcxx - Class messages<char> Function Interfaces

messages<char>::messages(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
messages<char>::messages(unsigned int)(GLIBCXX_3.4) [ISOCXX]
messages<char>::messages(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
messages<char>::messages(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.94 Class messages<wchar_t>

6.1.94.1 Class data for messages<wchar_t>

The virtual table for the std::messages<wchar_t> class is described in the generic part of this specification.

6.1.94.2 Interfaces for Class messages<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::messages<wchar_t> specified in Table 6-152, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-152 libstdcxx - Class messages<wchar_t> Function Interfaces

messages<wchar_t>::messages(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::messages(unsigned int)(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::messages(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
messages<wchar_t>::messages(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.95 Class messages_byname<char>

6.1.95.1 Class data for messages_byname<char>

The virtual table for the std::messages_byname<char> class is described in the generic part of this specification.

The Run Time Type Information for the std::messages_byname<char> class is described by Table 6-153

Table 6-153 typeinfo for messages_byname<char>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for messages_byname<char>

6.1.95.2 Interfaces for Class messages_byname<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::messages_byname<char> specified in Table 6-154, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-154 libstdcxx - Class messages_byname<char> Function Interfaces

messages_byname<char>::messages_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
messages_byname<char>::messages_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.96 Class messages_byname<wchar_t>

6.1.96.1 Class data for messages_byname<wchar_t>

The virtual table for the std::messages_byname<wchar_t> class is described in the generic part of this specification.

The Run Time Type Information for the std::messages_byname<wchar_t> class is described by Table 6-155

Table 6-155 typeinfo for messages_byname<wchar_t>

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

6.1.96.2 Interfaces for Class messages_byname<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::messages_byname<wchar_t> specified in Table 6-156, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-156 libstdcxx - Class messages_byname<wchar_t> Function Interfaces

messages_byname<wchar_t>::messages_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
messages_byname<wchar_t>::messages_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.97 Class numpunct<char>

6.1.97.1 Class data for numpunct<char>

The virtual table for the std::numpunct<char> class is described in the generic part of this specification.

The Run Time Type Information for the std::numpunct<char> class is described by Table 6-157

Table 6-157 typeinfo for numpunct<char>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for numpunct<char>

6.1.97.2 Interfaces for Class numpunct<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::numpunct<char> specified in Table 6-158, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-158 libstdcxx - Class numpunct<char> Function Interfaces

numpunct<char>::numpunct(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
numpunct<char>::numpunct(__numpunct_cache<char>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
numpunct<char>::numpunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]
numpunct<char>::numpunct(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
numpunct<char>::numpunct(__numpunct_cache<char>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

numpunct<char>::numpunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.98 Class numpunct<wchar_t>

6.1.98.1 Class data for numpunct<wchar_t>

The virtual table for the std::numpunct<wchar_t> class is described in the generic part of this specification.

The Run Time Type Information for the std::numpunct<wchar_t> class is described by Table 6-159

Table 6-159 typeinfo for numpunct<wchar_t>

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

6.1.98.2 Interfaces for Class numpunct<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::numpunct<wchar_t> specified in Table 6-160, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-160 libstdcxx - Class numpunct<wchar_t> Function Interfaces

numpunct<wchar_t>::numpunct(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::numpunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::numpunct(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
numpunct<wchar_t>::numpunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.99 Class numpunct_byname<char>

6.1.99.1 Class data for numpunct_byname<char>

The virtual table for the std::numpunct_byname<char> class is described in the generic part of this specification.

The Run Time Type Information for the std::numpunct_byname<char> class is described by Table 6-161

Table 6-161 typeinfo for numpunct_byname<char>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for numpunct_byname<char>

6.1.99.2 Interfaces for Class numpunct_byname<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::numpunct_byname<char> specified in Table 6-162, with

the full mandatory functionality as described in the referenced underlying specification.

Table 6-162 libstdcxx - Class numpunct_byname<char> Function Interfaces

numpunct_byname<char>::numpunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
numpunct_byname<char>::numpunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.100 Class numpunct_byname<wchar_t>

6.1.100.1 Class data for numpunct_byname<wchar_t>

The virtual table for the std::numpunct_byname<wchar_t> class is described in the generic part of this specification.

The Run Time Type Information for the std::numpunct_byname<wchar_t> class is described by Table 6-163

Table 6-163 typeinfo for numpunct_byname<wchar_t>

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

6.1.100.2 Interfaces for Class numpunct_byname<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::numpunct_byname<wchar_t> specified in Table 6-164, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-164 libstdcxx - Class numpunct_byname<wchar_t> Function Interfaces

numpunct_byname<wchar_t>::numpunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
numpunct_byname<wchar_t>::numpunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.101 Class __codecvt_abstract_base<wchar_t, char, __mbstate_t>

6.1.101.1 Class data for __codecvt_abstract_base<wchar_t, char, __mbstate_t>

The virtual table for the std::__codecvt_abstract_base<wchar_t, char, __mbstate_t> class is described in the generic part of this specification.

6.1.101.2 Interfaces for Class __codecvt_abstract_base<wchar_t, char, __mbstate_t>

No external methods are defined for libstdcxx - Class std::__codecvt_abstract_base<wchar_t, char, __mbstate_t> in this part of the specification. See also the generic specification.

6.1.102 Class codecvt_base

6.1.102.1 Class data for codecvt_base

The Run Time Type Information for the std::codecvt_base class is described by Table 6-165

Table 6-165 typeinfo for codecvt_base

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for codecvt_base

6.1.102.2 Interfaces for Class codecvt_base

No external methods are defined for libstdcxx - Class std::codecvt_base in this part of the specification. See also the generic specification.

6.1.103 Class codecvt<char, char, __mbstate_t>

6.1.103.1 Class data for codecvt<char, char, __mbstate_t>

The virtual table for the std::codecvt<char, char, __mbstate_t> class is described by Table 6-166

Table 6-166 Primary vtable for codecvt<char, char, __mbstate_t>

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for codecvt<char, char, __mbstate_t>
vfunc[0]:	codecvt<char, char, __mbstate_t>::~codecvt()
vfunc[1]:	codecvt<char, char, __mbstate_t>::~codecvt()
vfunc[2]:	codecvt<char, char, __mbstate_t>::do_out(__mbstate_t&, char const*, char const*, char const*&, char*, char*, char*&) const
vfunc[3]:	codecvt<char, char, __mbstate_t>::do_unshift(__mbstate_t&, char*, char*, char*&) const
vfunc[4]:	codecvt<char, char, __mbstate_t>::do_in(__mbstate_t&, char const*, char const*, char const*&, char*, char*, char*&) const
vfunc[5]:	codecvt<char, char, __mbstate_t>::do_encoding() const
vfunc[6]:	codecvt<char, char, __mbstate_t>::do_always_noconv() const

vfunc[7]:	codecvt<char, char, __mbstate_t>::do_length(__mbstate_t &, char const*, char const*, unsigned int) const
vfunc[8]:	codecvt<char, char, __mbstate_t>::do_max_length() const

The Run Time Type Information for the std::codecvt<char, char, __mbstate_t> class is described by Table 6-167

Table 6-167 typeinfo for codecvt<char, char, __mbstate_t>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for codecvt<char, char, __mbstate_t>

6.1.103.2 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 in the generic part of this specification.

6.1.103.3 Interfaces for Class codecvt<char, char, __mbstate_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::codecvt<char, char, __mbstate_t> specified in Table 6-168, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-168 libstdcxx - Class codecvt<char, char, __mbstate_t> Function Interfaces

codecvt<char, char, __mbstate_t>::do_length(__mbstate_t&, char const*, char const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
codecvt<char, char, __mbstate_t>::codecvt(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
codecvt<char, char, __mbstate_t>::codecvt(unsigned int)(GLIBCXX_3.4) [ISOCXX]
codecvt<char, char, __mbstate_t>::codecvt(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
codecvt<char, char, __mbstate_t>::codecvt(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.104 Class codecvt<wchar_t, char, __mbstate_t>

6.1.104.1 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 6-169

Table 6-169 Primary vtable for codecvt<wchar_t, char, __mbstate_t>

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for codecvt<wchar_t, char, __mbstate_t>
vfunc[0]:	codecvt<wchar_t, char, __mbstate_t>::~codecvt()
vfunc[1]:	codecvt<wchar_t, char, __mbstate_t>::~codecvt()
vfunc[2]:	codecvt<wchar_t, char, __mbstate_t>::do_out(__mbstate_t&, wchar_t const*, wchar_t const*, wchar_t const*&, char*, char*, char*&) const
vfunc[3]:	codecvt<wchar_t, char, __mbstate_t>::do_unshift(__mbstate_t&, char*, char*, char*&) const
vfunc[4]:	codecvt<wchar_t, char, __mbstate_t>::do_in(__mbstate_t&, char const*, char const*, char const*&, wchar_t*, wchar_t*, wchar_t*&) const
vfunc[5]:	codecvt<wchar_t, char, __mbstate_t>::do_encoding() const
vfunc[6]:	codecvt<wchar_t, char, __mbstate_t>::do_always_noconv() const
vfunc[7]:	codecvt<wchar_t, char, __mbstate_t>::do_length(__mbstate_t&, char const*, char const*, unsigned int) const
vfunc[8]:	codecvt<wchar_t, char, __mbstate_t>::do_max_length() const

The Run Time Type Information for the std::codecvt<wchar_t, char, __mbstate_t> class is described by Table 6-170

Table 6-170 typeinfo for codecvt<wchar_t, char, __mbstate_t>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for codecvt<wchar_t, char, __mbstate_t>

6.1.104.2 Interfaces for Class codecvt<wchar_t, char, __mbstate_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::codecvt<wchar_t, char, __mbstate_t> specified in Table

6-171, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-171 libstdcxx - Class codecvt<wchar_t, char, __mbstate_t> Function Interfaces

codecvt<wchar_t, char, __mbstate_t>::do_length(__mbstate_t&, char const*, char const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
codecvt<wchar_t, char, __mbstate_t>::codecvt(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
codecvt<wchar_t, char, __mbstate_t>::codecvt(unsigned int)(GLIBCXX_3.4) [ISOCXX]
codecvt<wchar_t, char, __mbstate_t>::codecvt(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
codecvt<wchar_t, char, __mbstate_t>::codecvt(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.105 Class codecvt_byname<char, char, __mbstate_t>

6.1.105.1 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 6-172

Table 6-172 Primary vtable for codecvt_byname<char, char, __mbstate_t>

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for codecvt_byname<char, char, __mbstate_t>
vfunc[0]:	codecvt_byname<char, char, __mbstate_t>::~codecvt_byname()
vfunc[1]:	codecvt_byname<char, char, __mbstate_t>::~codecvt_byname()
vfunc[2]:	codecvt<char, char, __mbstate_t>::do_out(__mbstate_t&, char const*, char const*, char const*&, char*, char*, char*&) const
vfunc[3]:	codecvt<char, char, __mbstate_t>::do_unshift(__mbstate_t&, char*, char*, char*&) const
vfunc[4]:	codecvt<char, char, __mbstate_t>::do_in(__mbstate_t&, char const*, char const*, char const*&, char*, char*, char*&) const
vfunc[5]:	codecvt<char, char, __mbstate_t>::do_encoding() const
vfunc[6]:	codecvt<char, char,

	__mbstate_t>::do_always_noconv() const
vfunc[7]:	codecvt<char, char, __mbstate_t>::do_length(__mbstate_t &, char const*, char const*, unsigned int) const
vfunc[8]:	codecvt<char, char, __mbstate_t>::do_max_length() const

The Run Time Type Information for the std::codecvt_byname<char, char, __mbstate_t> class is described by Table 6-173

Table 6-173 typeinfo for codecvt_byname<char, char, __mbstate_t>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for codecvt_byname<char, char, __mbstate_t>

6.1.105.2 Interfaces for Class codecvt_byname<char, char, __mbstate_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::codecvt_byname<char, char, __mbstate_t> specified in Table 6-174, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-174 libstdcxx - Class codecvt_byname<char, char, __mbstate_t> Function Interfaces

codecvt_byname<char, char, __mbstate_t>::codecvt_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
codecvt_byname<char, char, __mbstate_t>::codecvt_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.106 Class codecvt_byname<wchar_t, char, __mbstate_t>

6.1.106.1 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 6-175

Table 6-175 Primary vtable for codecvt_byname<wchar_t, char, __mbstate_t>

Base Offset	0
Virtual Base Offset	0
RTTI	typeinfo for codecvt_byname<wchar_t, char, __mbstate_t>
vfunc[0]:	codecvt_byname<wchar_t, char,

	__mbstate_t>::~codecvt_byname()
vfunc[1]:	codecvt_byname<wchar_t, char, __mbstate_t>::~codecvt_byname()
vfunc[2]:	codecvt<wchar_t, char, __mbstate_t>::do_out(__mbstate_t&, wchar_t const*, wchar_t const*, wchar_t const*&, char*, char*, char*&) const
vfunc[3]:	codecvt<wchar_t, char, __mbstate_t>::do_unshift(__mbstate_t&, char*, char*, char*&) const
vfunc[4]:	codecvt<wchar_t, char, __mbstate_t>::do_in(__mbstate_t&, char const*, char const*, char const*&, wchar_t*, wchar_t*, wchar_t*&) const
vfunc[5]:	codecvt<wchar_t, char, __mbstate_t>::do_encoding() const
vfunc[6]:	codecvt<wchar_t, char, __mbstate_t>::do_always_noconv() const
vfunc[7]:	codecvt<wchar_t, char, __mbstate_t>::do_length(__mbstate_t&, char const*, char const*, unsigned int) const
vfunc[8]:	codecvt<wchar_t, char, __mbstate_t>::do_max_length() const

The Run Time Type Information for the std::codecvt_byname<wchar_t, char, __mbstate_t> class is described by Table 6-176

Table 6-176 typeinfo for codecvt_byname<wchar_t, char, __mbstate_t>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for codecvt_byname<wchar_t, char, __mbstate_t>

6.1.106.2 Class data for collate_byname<wchar_t>

The virtual table for the std::collate_byname<wchar_t> class is described in the generic part of this specification.

The Run Time Type Information for the std::collate_byname<wchar_t> class is described by Table 6-177

Table 6-177 typeinfo for collate_byname<wchar_t>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info

Name	typeinfo name for collate_byname<wchar_t>

6.1.106.3 Interfaces for Class codecvt_byname<wchar_t, char, __mbstate_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::codecvt_byname<wchar_t, char, __mbstate_t> specified in Table 6-178, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-178 libstdcxx - Class codecvt_byname<wchar_t, char, __mbstate_t> Function Interfaces

codecvt_byname<wchar_t, char, __mbstate_t>::codecvt_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
codecvt_byname<wchar_t, char, __mbstate_t>::codecvt_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
collate_byname<wchar_t>::collate_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
collate_byname<wchar_t>::collate_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.107 Class collate<char>

6.1.107.1 Class data for collate<char>

The virtual table for the std::collate<char> class is described in the generic part of this specification.

The Run Time Type Information for the std::collate<char> class is described by Table 6-179

Table 6-179 typeinfo for collate<char>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for collate<char>

6.1.107.2 Interfaces for Class collate<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::collate<char> specified in Table 6-180, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-180 libstdcxx - Class collate<char> Function Interfaces

collate<char>::_M_transform(char*, char const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
collate<char>::collate(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
collate<char>::collate(unsigned int)(GLIBCXX_3.4) [ISOCXX]
collate<char>::collate(__locale_struct*, unsigned int)(GLIBCXX_3.4)

[ISOCXX]
collate<char>::collate(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.108 Class collate<wchar_t>

6.1.108.1 Class data for collate<wchar_t>

The virtual table for the std::collate<wchar_t> class is described in the generic part of this specification.

The Run Time Type Information for the std::collate<wchar_t> class is described by Table 6-181

Table 6-181 typeinfo for collate<wchar_t>

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

6.1.108.2 Interfaces for Class collate<wchar_t>

An LSB conforming implementation shall provide the architecture specific methods for Class std::collate<wchar_t> specified in Table 6-182, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-182 libstdcxx - Class collate<wchar_t> Function Interfaces

collate<wchar_t>::_M_transform(wchar_t*, wchar_t const*, unsigned int) const(GLIBCXX_3.4) [ISOCXX]
collate<wchar_t>::collate(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
collate<wchar_t>::collate(unsigned int)(GLIBCXX_3.4) [ISOCXX]
collate<wchar_t>::collate(__locale_struct*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
collate<wchar_t>::collate(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.109 Class collate_byname<char>

6.1.109.1 Class data for collate_byname<char>

The virtual table for the std::collate_byname<char> class is described in the generic part of this specification.

The Run Time Type Information for the std::collate_byname<char> class is described by Table 6-183

Table 6-183 typeinfo for collate_byname<char>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for collate_byname<char>

6.1.109.2 Interfaces for Class collate_byname<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::collate_byname<char> specified in Table 6-184, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-184 libstdcxx - Class collate_byname<char> Function Interfaces

collate_byname<char>::collate_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
collate_byname<char>::collate_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.110 Class time_base

6.1.110.1 Class data for time_base

The Run Time Type Information for the std::time_base class is described by Table 6-185

Table 6-185 typeinfo for time_base

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for time_base

6.1.110.2 Interfaces for Class time_base

No external methods are defined for libstdcxx - Class std::time_base in this part of the specification. See also the generic specification.

6.1.111 Class time_get_byname<char, istreambuf_iterator<char, char_traits<char> > >

6.1.111.1 Class data for time_get_byname<char, istreambuf_iterator<char, char_traits<char> > >

The virtual table for the std::time_get_byname<char, std::istreambuf_iterator<char, std::char_traits<char> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::time_get_byname<char, std::istreambuf_iterator<char, std::char_traits<char> > > class is described by Table 6-186

Table 6-186 typeinfo for time_get_byname<char, istreambuf_iterator<char, char_traits<char> > >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for time_get_byname<char, istreambuf_iterator<char, char_traits<char> > >

6.1.111.2 Interfaces for Class time_get_byname<char, istreambuf_iterator<char, char_traits<char> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::time_get_byname<char, std::istreambuf_iterator<char, std::char_traits<char> > > specified in Table 6-187, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-187 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(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
time_get_byname<char, istreambuf_iterator<char, char_traits<char> > >::time_get_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.112 Class time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

6.1.112.1 Class data for time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

The virtual table for the std::time_get_byname<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::time_get_byname<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described by Table 6-188

Table 6-188 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> > >

6.1.112.2 Interfaces for Class time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::time_get_byname<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 6-189, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-189 libstdcxx - Class time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > Function Interfaces

time_get_byname<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::time_get_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
time_get_byname<wchar_t, istreambuf_iterator<wchar_t,

char_traits<wchar_t> > >::time_get_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.113 Class time_put_byname<char, ostreambuf_iterator<char, char_traits<char> > >

6.1.113.1 Class data for time_put_byname<char, ostreambuf_iterator<char, char_traits<char> > >

The virtual table for the std::time_put_byname<char, std::ostreambuf_iterator<char, std::char_traits<char> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::time_put_byname<char, std::ostreambuf_iterator<char, std::char_traits<char> > > class is described by Table 6-190

Table 6-190 typeinfo for time_put_byname<char, ostreambuf_iterator<char, char_traits<char> > >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for time_put_byname<char, ostreambuf_iterator<char, char_traits<char> > >

6.1.113.2 Interfaces for Class time_put_byname<char, ostreambuf_iterator<char, char_traits<char> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::time_put_byname<char, std::ostreambuf_iterator<char, std::char_traits<char> > > specified in Table 6-191, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-191 libstdcxx - Class time_put_byname<char, ostreambuf_iterator<char, char_traits<char> > > Function Interfaces

time_put_byname<char, ostreambuf_iterator<char, char_traits<char> > >::time_put_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
time_put_byname<char, ostreambuf_iterator<char, char_traits<char> > >::time_put_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.114 Class time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

6.1.114.1 Class data for time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

The virtual table for the std::time_put_byname<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::time_put_byname<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described by Table 6-192

Table 6-192 typeinfo for time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

6.1.114.2 Interfaces for Class time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::time_put_byname<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 6-193, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-193 libstdcxx - Class time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > Function Interfaces

time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::time_put_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
time_put_byname<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::time_put_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.115 Class time_get<char, istreambuf_iterator<char, char_traits<char> > >

6.1.115.1 Class data for time_get<char, istreambuf_iterator<char, char_traits<char> > >

The virtual table for the std::time_get<char, std::istreambuf_iterator<char, std::char_traits<char> > > class is described in the generic part of this specification.

6.1.115.2 Interfaces for Class time_get<char, istreambuf_iterator<char, char_traits<char> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::time_get<char, std::istreambuf_iterator<char, std::char_traits<char> > > specified in Table 6-194, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-194 libstdcxx - Class time_get<char, istreambuf_iterator<char, char_traits<char> > > Function Interfaces

time_get<char, istreambuf_iterator<char, char_traits<char> > >::_M_extract_num(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, int&, int, int, unsigned int, ios_base&, _Ios_Iostate&) const(GLIBCXX_3.4) [ISOCXX]
time_get<char, istreambuf_iterator<char, char_traits<char> > >

>::_M_extract_name(istreambuf_iterator<char, char_traits<char> >, istreambuf_iterator<char, char_traits<char> >, int&, char const**, unsigned int, ios_base&, _Ios_Iostate&) const(GLIBCXX_3.4) [ISOCXX]
time_get<char, istreambuf_iterator<char, char_traits<char> > >::time_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]
time_get<char, istreambuf_iterator<char, char_traits<char> > >::time_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.116 Class time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

6.1.116.1 Class data for time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

The virtual table for the std::time_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described in the generic part of this specification.

6.1.116.2 Interfaces for Class time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::time_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 6-195, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-195 libstdcxx - Class time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > Function Interfaces

time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::_M_extract_num(istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, int&, int, int, unsigned int, ios_base&, _Ios_Iostate&) const(GLIBCXX_3.4) [ISOCXX]
time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::_M_extract_name(istreambuf_iterator<wchar_t, char_traits<wchar_t> >, istreambuf_iterator<wchar_t, char_traits<wchar_t> >, int&, wchar_t const**, unsigned int, ios_base&, _Ios_Iostate&) const(GLIBCXX_3.4) [ISOCXX]
time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::time_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]
time_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::time_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.117 Class time_put<char, ostreambuf_iterator<char, char_traits<char> > >

6.1.117.1 Class data for time_put<char, ostreambuf_iterator<char, char_traits<char> > >

The virtual table for the std::time_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::time_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > class is described by Table 6-196

Table 6-196 typeinfo for time_put<char, ostreambuf_iterator<char, char_traits<char> > >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info	
Name	typeinfo name for time_put<char, ostreambuf_iterator<char, char_traits<char> > >	
flags:	8	
basetype:	typeinfo for locale::facet	2
basetype:	typeinfo for time_base	2

6.1.117.2 Interfaces for Class time_put<char, ostreambuf_iterator<char, char_traits<char> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::time_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > specified in Table 6-197, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-197 libstdcxx - Class time_put<char, ostreambuf_iterator<char, char_traits<char> > > Function Interfaces

time_put<char, ostreambuf_iterator<char, char_traits<char> > >::time_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]
time_put<char, ostreambuf_iterator<char, char_traits<char> > >::time_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.118 Class time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

6.1.118.1 Class data for time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

The virtual table for the std::time_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::time_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described by Table 6-198

Table 6-198 typeinfo for time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

Base Vtable	vtable for __cxxabiv1::__si_class_t	

	ype_info	
Name	typeinfo name for time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >	
flags:	8	
basetype:	typeinfo for locale::facet	2
basetype:	typeinfo for time_base	2

6.1.118.2 Interfaces for Class time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::time_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 6-199, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-199 libstdcxx - Class time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > Function Interfaces

time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::time_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]
time_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::time_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.119 Class moneypunct<char, false>

6.1.119.1 Class data for moneypunct<char, false>

The virtual table for the std::moneypunct<char, false> class is described in the generic part of this specification.

6.1.119.2 Interfaces for Class moneypunct<char, false>

An LSB conforming implementation shall provide the architecture specific methods for Class std::moneypunct<char, false> specified in Table 6-200, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-200 libstdcxx - Class moneypunct<char, false> Function Interfaces

moneypunct<char, false>::moneypunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<char, false>::moneypunct(__moneypunct_cache<char, false>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<char, false>::moneypunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<char, false>::moneypunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<char, false>::moneypunct(__moneypunct_cache<char, false>*,

unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<char, false>::moneypunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.120 Class moneypunct<char, true>

6.1.120.1 Class data for moneypunct<char, true>

The virtual table for the std::moneypunct<char, true> class is described in the generic part of this specification.

6.1.120.2 Interfaces for Class moneypunct<char, true>

An LSB conforming implementation shall provide the architecture specific methods for Class std::moneypunct<char, true> specified in Table 6-201, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-201 libstdcxx - Class moneypunct<char, true> Function Interfaces

moneypunct<char, true>::moneypunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<char, true>::moneypunct(__moneypunct_cache<char, true>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<char, true>::moneypunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<char, true>::moneypunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<char, true>::moneypunct(__moneypunct_cache<char, true>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<char, true>::moneypunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.121 Class moneypunct<wchar_t, false>

6.1.121.1 Class data for moneypunct<wchar_t, false>

The virtual table for the std::moneypunct<wchar_t, false> class is described in the generic part of this specification.

6.1.121.2 Interfaces for Class moneypunct<wchar_t, false>

An LSB conforming implementation shall provide the architecture specific methods for Class std::moneypunct<wchar_t, false> specified in Table 6-202, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-202 libstdcxx - Class moneypunct<wchar_t, false> Function Interfaces

moneypunct<wchar_t, false>::moneypunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<wchar_t, false>::moneypunct(__moneypunct_cache<wchar_t, false>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

moneypunct<wchar_t, false>::moneypunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<wchar_t, false>::moneypunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<wchar_t, false>::moneypunct(__moneypunct_cache<wchar_t, false>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<wchar_t, false>::moneypunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.122 Class moneypunct<wchar_t, true>

6.1.122.1 Class data for moneypunct<wchar_t, true>

The virtual table for the std::moneypunct<wchar_t, true> class is described in the generic part of this specification.

6.1.122.2 Interfaces for Class moneypunct<wchar_t, true>

An LSB conforming implementation shall provide the architecture specific methods for Class std::moneypunct<wchar_t, true> specified in Table 6-203, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-203 libstdcxx - Class moneypunct<wchar_t, true> Function Interfaces

moneypunct<wchar_t, true>::moneypunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<wchar_t, true>::moneypunct(__moneypunct_cache<wchar_t, true>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<wchar_t, true>::moneypunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<wchar_t, true>::moneypunct(__locale_struct*, char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<wchar_t, true>::moneypunct(__moneypunct_cache<wchar_t, true>*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct<wchar_t, true>::moneypunct(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.123 Class moneypunct_byname<char, false>

6.1.123.1 Class data for moneypunct_byname<char, false>

The virtual table for the std::moneypunct_byname<char, false> class is described in the generic part of this specification.

The Run Time Type Information for the std::moneypunct_byname<char, false> class is described by Table 6-204

Table 6-204 typeinfo for moneypunct_byname<char, false>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info

Name	typeinfo name for moneypunct_byname<char, false>

6.1.123.2 Interfaces for Class moneypunct_byname<char, false>

An LSB conforming implementation shall provide the architecture specific methods for Class std::moneypunct_byname<char, false> specified in Table 6-205, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-205 libstdcxx - Class moneypunct_byname<char, false> Function Interfaces

moneypunct_byname<char, false>::moneypunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct_byname<char, false>::moneypunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.124 Class moneypunct_byname<char, true>

6.1.124.1 Class data for moneypunct_byname<char, true>

The virtual table for the std::moneypunct_byname<char, true> class is described in the generic part of this specification.

The Run Time Type Information for the std::moneypunct_byname<char, true> class is described by Table 6-206

Table 6-206 typeinfo for moneypunct_byname<char, true>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for moneypunct_byname<char, true>

6.1.124.2 Interfaces for Class moneypunct_byname<char, true>

An LSB conforming implementation shall provide the architecture specific methods for Class std::moneypunct_byname<char, true> specified in Table 6-207, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-207 libstdcxx - Class moneypunct_byname<char, true> Function Interfaces

moneypunct_byname<char, true>::moneypunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct_byname<char, true>::moneypunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.125 Class moneypunct_byname<wchar_t, false>

6.1.125.1 Class data for moneypunct_byname<wchar_t, false>

The virtual table for the std::moneypunct_byname<wchar_t, false> class is described in the generic part of this specification.

The Run Time Type Information for the std::moneypunct_byname<wchar_t, false> class is described by Table 6-208

Table 6-208 typeinfo for moneypunct_byname<wchar_t, false>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for moneypunct_byname<wchar_t, false>

6.1.125.2 Interfaces for Class moneypunct_byname<wchar_t, false>

An LSB conforming implementation shall provide the architecture specific methods for Class std::moneypunct_byname<wchar_t, false> specified in Table 6-209, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-209 libstdcxx - Class moneypunct_byname<wchar_t, false> Function Interfaces

moneypunct_byname<wchar_t, false>::moneypunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]
moneypunct_byname<wchar_t, false>::moneypunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.126 Class moneypunct_byname<wchar_t, true>

6.1.126.1 Class data for moneypunct_byname<wchar_t, true>

The virtual table for the std::moneypunct_byname<wchar_t, true> class is described in the generic part of this specification.

The Run Time Type Information for the std::moneypunct_byname<wchar_t, true> class is described by Table 6-210

Table 6-210 typeinfo for moneypunct_byname<wchar_t, true>

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for moneypunct_byname<wchar_t, true>

6.1.126.2 Interfaces for Class moneypunct_byname<wchar_t, true>

An LSB conforming implementation shall provide the architecture specific methods for Class std::moneypunct_byname<wchar_t, true> specified in Table 6-211, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-211 libstdcxx - Class moneypunct_byname<wchar_t, true> Function Interfaces

moneypunct_byname<wchar_t, true>::moneypunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

moneypunct_byname<wchar_t, true>::moneypunct_byname(char const*, unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.127 Class money_base

6.1.127.1 Class data for money_base

The Run Time Type Information for the std::money_base class is described by Table 6-212

Table 6-212 typeinfo for money_base

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for money_base

6.1.127.2 Interfaces for Class money_base

No external methods are defined for libstdcxx - Class std::money_base in this part of the specification. See also the generic specification.

6.1.128 Class money_get<char, istreambuf_iterator<char, char_traits<char> > >

6.1.128.1 Class data for money_get<char, istreambuf_iterator<char, char_traits<char> > >

The virtual table for the std::money_get<char, std::istreambuf_iterator<char, std::char_traits<char> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::money_get<char, std::istreambuf_iterator<char, std::char_traits<char> > > class is described by Table 6-213

Table 6-213 typeinfo for money_get<char, istreambuf_iterator<char, char_traits<char> > >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for money_get<char, istreambuf_iterator<char, char_traits<char> > >

6.1.128.2 Interfaces for Class money_get<char, istreambuf_iterator<char, char_traits<char> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::money_get<char, std::istreambuf_iterator<char, std::char_traits<char> > > specified in Table 6-214, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-214 libstdcxx - Class money_get<char, istreambuf_iterator<char, char_traits<char> > > Function Interfaces

money_get<char, istreambuf_iterator<char, char_traits<char> > >::money_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]

money_get<char, istreambuf_iterator<char, char_traits<char> > >::money_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.129 Class money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

6.1.129.1 Class data for money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

The virtual table for the std::money_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::money_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described by Table 6-215

Table 6-215 typeinfo for money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

6.1.129.2 Interfaces for Class money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::money_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 6-216, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-216 libstdcxx - Class money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > Function Interfaces

money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::money_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]
money_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::money_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.130 Class money_put<char, ostreambuf_iterator<char, char_traits<char> > >

6.1.130.1 Class data for money_put<char, ostreambuf_iterator<char, char_traits<char> > >

The virtual table for the std::money_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::money_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > class is described by Table 6-217

Table 6-217 typeinfo for money_put<char, ostreambuf_iterator<char, char_traits<char> > >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for money_put<char, ostreambuf_iterator<char, char_traits<char> > >

6.1.130.2 Interfaces for Class money_put<char, ostreambuf_iterator<char, char_traits<char> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::money_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > specified in Table 6-218, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-218 libstdcxx - Class money_put<char, ostreambuf_iterator<char, char_traits<char> > > Function Interfaces

money_put<char, ostreambuf_iterator<char, char_traits<char> > >::money_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]
money_put<char, ostreambuf_iterator<char, char_traits<char> > >::money_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.131 Class money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

6.1.131.1 Class data for money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

The virtual table for the std::money_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::money_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described by Table 6-219

Table 6-219 typeinfo for money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

6.1.131.2 Interfaces for Class money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::money_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 6-220, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-220 libstdcxx - Class money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > Function Interfaces

money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::money_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]
money_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::money_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.132 Class locale

6.1.132.1 Interfaces for Class locale

An LSB conforming implementation shall provide the architecture specific methods for Class std::locale specified in Table 6-221, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-221 libstdcxx - Class locale Function Interfaces

locale::_Impl::_Impl(char const*, unsigned int)(GLIBCXX_3.4) [LSB]
locale::_Impl::_Impl(locale::_Impl const&, unsigned int)(GLIBCXX_3.4) [LSB]
locale::_Impl::_Impl(unsigned int)(GLIBCXX_3.4) [LSB]
locale::_Impl::_Impl(char const*, unsigned int)(GLIBCXX_3.4) [LSB]
locale::_Impl::_Impl(locale::_Impl const&, unsigned int)(GLIBCXX_3.4) [LSB]
locale::_Impl::_Impl(unsigned int)(GLIBCXX_3.4) [LSB]

6.1.133 Class locale::facet

6.1.133.1 Class data for locale::facet

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

The Run Time Type Information for the std::locale::facet class is described by Table 6-222

Table 6-222 typeinfo for locale::facet

Base Vtable	vtable for __cxxabiv1::__class_type_info
Name	typeinfo name for locale::facet

6.1.133.2 Interfaces for Class locale::facet

No external methods are defined for libstdcxx - Class std::locale::facet in this part of the specification. See also the generic specification.

6.1.134 facet functions

6.1.134.1 Interfaces for facet functions

No external methods are defined for libstdcxx - facet functions in this part of the specification. See also the generic specification.

6.1.135 Class __num_base

6.1.135.1 Class data for __num_base

6.1.135.2 Interfaces for Class __num_base

No external methods are defined for libstdcxx - Class std::__num_base in this part of the specification. See also the generic specification.

6.1.136 Class num_get<char, istreambuf_iterator<char, char_traits<char> > >

6.1.136.1 Class data for num_get<char, istreambuf_iterator<char, char_traits<char> > >

The virtual table for the std::num_get<char, std::istreambuf_iterator<char, std::char_traits<char> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::num_get<char, std::istreambuf_iterator<char, std::char_traits<char> > > class is described by Table 6-223

Table 6-223 typeinfo for num_get<char, istreambuf_iterator<char, char_traits<char> > >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for num_get<char, istreambuf_iterator<char, char_traits<char> > >
basetype:	typeinfo for locale::facet

6.1.136.2 Interfaces for Class num_get<char, istreambuf_iterator<char, char_traits<char> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::num_get<char, std::istreambuf_iterator<char, std::char_traits<char> > > specified in Table 6-224, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-224 libstdcxx - Class num_get<char, istreambuf_iterator<char, char_traits<char> > > Function Interfaces

num_get<char, istreambuf_iterator<char, char_traits<char> > >::num_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]
num_get<char, istreambuf_iterator<char, char_traits<char> > >::num_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.137 Class num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

6.1.137.1 Class data for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

The virtual table for the std::num_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::num_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described by Table 6-225

Table 6-225 typeinfo for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >
basetype:	typeinfo for locale::facet

6.1.137.2 Interfaces for Class num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::num_get<wchar_t, std::istreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 6-226, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-226 libstdcxx - Class num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > > Function Interfaces

num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::num_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]
num_get<wchar_t, istreambuf_iterator<wchar_t, char_traits<wchar_t> > >::num_get(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.138 Class num_put<char, ostreambuf_iterator<char, char_traits<char> > >

6.1.138.1 Class data for num_put<char, ostreambuf_iterator<char, char_traits<char> > >

The virtual table for the std::num_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::num_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > class is described by Table 6-227

Table 6-227 typeinfo for num_put<char, ostreambuf_iterator<char, char_traits<char> > >

Base Vtable	vtable for __cxxabiv1::__si_class_type_info
Name	typeinfo name for num_put<char, ostreambuf_iterator<char, char_traits<char> > >
basetype:	typeinfo for locale::facet

6.1.138.2 Interfaces for Class num_put<char, ostreambuf_iterator<char, char_traits<char> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::num_put<char, std::ostreambuf_iterator<char, std::char_traits<char> > > specified in Table 6-228, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-228 libstdcxx - Class num_put<char, ostreambuf_iterator<char, char_traits<char> > > Function Interfaces

num_put<char, ostreambuf_iterator<char, char_traits<char> > >::_M_group_int(char const*, unsigned int, char, ios_base&, char*, char*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<char, ostreambuf_iterator<char, char_traits<char> > >::_M_group_float(char const*, unsigned int, char, char const*, char*, char*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<char, ostreambuf_iterator<char, char_traits<char> > >::_M_pad(char, int, ios_base&, char*, char const*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<char, ostreambuf_iterator<char, char_traits<char> > >::num_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]
num_put<char, ostreambuf_iterator<char, char_traits<char> > >::num_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.139 Class num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

6.1.139.1 Class data for num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

The virtual table for the std::num_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described in the generic part of this specification.

The Run Time Type Information for the std::num_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > class is described by Table 6-229

Table 6-229 typeinfo for num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

Base Vtable	vtable for

	__cxxabiv1::__si_class_type_info
Name	typeinfo name for num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >
basetype:	typeinfo for locale::facet

6.1.139.2 Interfaces for Class num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >

An LSB conforming implementation shall provide the architecture specific methods for Class std::num_put<wchar_t, std::ostreambuf_iterator<wchar_t, std::char_traits<wchar_t> > > specified in Table 6-230, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-230 libstdcxx - Class num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > > Function Interfaces

num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::_M_group_int(char const*, unsigned int, wchar_t, ios_base&, wchar_t*, wchar_t*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::_M_group_float(char const*, unsigned int, wchar_t, wchar_t const*, wchar_t*, wchar_t*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::_M_pad(wchar_t, int, ios_base&, wchar_t*, wchar_t const*, int&) const(GLIBCXX_3.4) [ISOCXX]
num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::num_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]
num_put<wchar_t, ostreambuf_iterator<wchar_t, char_traits<wchar_t> > >::num_put(unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.140 Class gslice

6.1.140.1 Class data for gslice

6.1.140.2 Interfaces for Class gslice

An LSB conforming implementation shall provide the architecture specific methods for Class std::gslice specified in Table 6-231, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-231 libstdcxx - Class gslice Function Interfaces

gslice::_Indexer::_Indexer(unsigned int, valarray<unsigned int> const&, valarray<unsigned int> const&)(GLIBCXX_3.4) [ISOCXX]
gslice::_Indexer::_Indexer(unsigned int, valarray<unsigned int> const&, valarray<unsigned int> const&)(GLIBCXX_3.4) [ISOCXX]

6.1.141 Class __basic_file<char>

6.1.141.1 Class data for __basic_file<char>

6.1.141.2 Interfaces for Class __basic_file<char>

An LSB conforming implementation shall provide the architecture specific methods for Class std::__basic_file<char> specified in Table 6-232, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-232 libstdcxx - Class __basic_file<char> Function Interfaces

__basic_file<char>::xsgetn(char*, int)(GLIBCXX_3.4) [ISOCXX]
__basic_file<char>::xsputn(char const*, int)(GLIBCXX_3.4) [ISOCXX]
__basic_file<char>::seekoff(long long, _Ios_Seekdir)(GLIBCXX_3.4) [ISOCXX]
__basic_file<char>::xsputn_2(char const*, int, char const*, int)(GLIBCXX_3.4) [ISOCXX]

6.1.142 Class _List_node_base

6.1.142.1 Interfaces for Class _List_node_base

No external methods are defined for libstdcxx - Class std::_List_node_base in this part of the specification. See also the generic specification.

6.1.143 Class valarray<unsigned int>

6.1.143.1 Class data for valarray<unsigned int>

6.1.143.2 Interfaces for Class valarray<unsigned int>

An LSB conforming implementation shall provide the architecture specific methods for Class std::valarray<unsigned int> specified in Table 6-233, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-233 libstdcxx - Class valarray<unsigned int> Function Interfaces

valarray<unsigned int>::size() const(GLIBCXX_3.4) [ISOCXX]
valarray<unsigned int>::valarray(valarray<unsigned int> const&)(GLIBCXX_3.4) [ISOCXX]
valarray<unsigned int>::valarray(unsigned int)(GLIBCXX_3.4) [ISOCXX]
valarray<unsigned int>::valarray(valarray<unsigned int> const&)(GLIBCXX_3.4) [ISOCXX]
valarray<unsigned int>::valarray(unsigned int)(GLIBCXX_3.4) [ISOCXX]
valarray<unsigned int>::~valarray()(GLIBCXX_3.4) [ISOCXX]
valarray<unsigned int>::~valarray()(GLIBCXX_3.4) [ISOCXX]
valarray<unsigned int>::operator[](unsigned int)(GLIBCXX_3.4) [ISOCXX]

6.1.144 Class allocator<char>

6.1.144.1 Class data for allocator<char>

6.1.144.2 Interfaces for Class allocator<char>

No external methods are defined for libstdcxx - Class std::allocator<char> in this part of the specification. See also the generic specification.

6.1.145 Class allocator<wchar_t>

6.1.145.1 Class data for allocator<wchar_t>

6.1.145.2 Interfaces for Class allocator<wchar_t>

No external methods are defined for libstdcxx - Class std::allocator<wchar_t> in this part of the specification. See also the generic specification.

6.1.146 Class __gnu_cxx::__pool<true>

6.1.146.1 Interfaces for Class __gnu_cxx::__pool<true>

An LSB conforming implementation shall provide the architecture specific methods for Class __gnu_cxx::__pool<true> specified in Table 6-234, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-234 libstdcxx - Class __gnu_cxx::__pool<true> Function Interfaces

__gnu_cxx::__pool<true>::_M_reclaim_block(char*, unsigned int)(GLIBCXX_3.4.4) [LSB]
__gnu_cxx::__pool<true>::_M_reserve_block(unsigned int, unsigned int)(GLIBCXX_3.4.4) [LSB]

6.1.147 Class __gnu_cxx::__pool<false>

6.1.147.1 Interfaces for Class __gnu_cxx::__pool<false>

An LSB conforming implementation shall provide the architecture specific methods for Class __gnu_cxx::__pool<false> specified in Table 6-235, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-235 libstdcxx - Class __gnu_cxx::__pool<false> Function Interfaces

__gnu_cxx::__pool<false>::_M_reclaim_block(char*, unsigned int)(GLIBCXX_3.4.4) [LSB]
__gnu_cxx::__pool<false>::_M_reserve_block(unsigned int, unsigned int)(GLIBCXX_3.4.4) [LSB]

6.1.148 Class __gnu_cxx::free_list

6.1.148.1 Interfaces for Class __gnu_cxx::free_list

An LSB conforming implementation shall provide the architecture specific methods for Class __gnu_cxx::free_list specified in Table 6-236, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-236 libstdcxx - Class __gnu_cxx::free_list Function Interfaces

__gnu_cxx::free_list::_M_get(unsigned int)(GLIBCXX_3.4.4) [LSB]

6.1.149 Class locale::_Impl

6.1.149.1 Interfaces for Class locale::_Impl

An LSB conforming implementation shall provide the architecture specific methods for Class std::locale::_Impl specified in Table 6-237, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-237 libstdcxx - Class locale::_Impl Function Interfaces

locale::_Impl::_M_install_cache(locale::facet const*, unsigned int)(GLIBCXX_3.4.7) [ISOCXX]

6.1.150 Namespace std Functions

6.1.150.1 Interfaces for Namespace std Functions

An LSB conforming implementation shall provide the architecture specific methods for Namespace std Functions specified in Table 6-238, with the full mandatory functionality as described in the referenced underlying specification.

Table 6-238 libstdcxx - Namespace std Functions Function Interfaces

int __copy_streambufs<char, char_traits<char> >(basic_streambuf<char, char_traits<char> >*, basic_streambuf<char, char_traits<char> >*)(GLIBCXX_3.4.6) [ISOCXX]
int __copy_streambufs<wchar_t, char_traits<wchar_t> >(basic_streambuf<wchar_t, char_traits<wchar_t> >*, basic_streambuf<wchar_t, char_traits<wchar_t> >*)(GLIBCXX_3.4.6) [ISOCXX]

6.1.151 Class char_traits<char>

6.1.151.1 Interfaces for Class char_traits<char>

No external methods are defined for libstdcxx - Class std::char_traits<char> in this part of the specification. See also the generic specification.

6.1.152 Class char_traits<wchar_t>

6.1.152.1 Interfaces for Class char_traits<wchar_t>

No external methods are defined for libstdcxx - Class std::char_traits<wchar_t> in this part of the specification. See also the generic specification.

6.2 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 6.1 shall behave as described in the referenced base document. For interfaces referencing LSB and not listed below, please see the generic part of the specification.

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. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

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.