

Linux Standard Base Core Specification for AMD64 2.0.1

Linux Standard Base Core Specification for AMD64 2.0.1

Copyright © 2004 Free Standards Group

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

Portions of the text are copyrighted by the following parties:

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

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

Linux is a trademark of Linus Torvalds.

UNIX a registered trademark of the Open Group in the United States and other countries.

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

AMD is a trademark of Advanced Micro Devices, Inc.

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

OpenGL is a registered trademark of Silicon Graphics, Inc.

Specification Introduction

Specification Introduction

Table of Contents

Foreword	i
Introduction	ii
I. Introductory Elements	3
1. Scope	1
1.1. General	1
1.2. Module Specific Scope	1
2. Normative References	2
3. Requirements	5
3.1. Relevant Libraries	5
3.2. LSB Implementation Conformance	5
3.3. LSB Application Conformance	6
4. Definitions	7
5. Terminology	8
6. Documentation Conventions	9

List of Tables

2-1. Normative References	2
3-1. Standard Library Names	5

Foreword

- 1 This is version 2.0.1 of the Linux Standard Base Core Specification for AMD64. An implementation of this version of
- 2 the specification may not claim to be an implementation of the Linux Standard Base unless it has successfully
- 3 completed the compliance process as defined by the Free Standards Group.

Introduction

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

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

I. Introductory Elements

Chapter 1. Scope

1.1. General

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

4 These specifications are composed of two basic parts: A common specification ("LSB-generic") describing those parts
5 of the interface that remain constant across all implementations of the LSB, and an architecture-specific specification
6 ("LSB-arch") describing the parts of the interface that vary by processor architecture. Together, the LSB-generic and
7 the architecture-specific supplement for a single hardware architecture provide a complete interface specification for
8 compiled application programs on systems that share a common hardware architecture.

9 The LSB-generic document shall be used in conjunction with an architecture-specific supplement. Whenever a section
10 of the LSB-generic specification shall be supplemented by architecture-specific information, the LSB-generic
11 document includes a reference to the architecture supplement. Architecture supplements may also contain additional
12 information that is not referenced in the LSB-generic document.

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

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

1.2. Module Specific Scope

20 This is the AMD64 architecture specific Core module of the Linux Standards Base (LSB). This module supplements
21 the generic LSB Core module with those interfaces that differ between architectures.

22 Interfaces described in this module are mandatory except where explicitly listed otherwise. Core interfaces may be
23 supplemented by other modules; all modules are built upon the core.

Chapter 2. Normative References

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

4 **Table 2-1. Normative References**

Name	Title	URL
AMD64 Architecture Programmer's Manual, Volume 1	AMD64 Architecture Programmer's Manual, Volume 1: Application Programming 24592 3.08	http://www.amd.com/us-en/Processors/DevelopWithAMD/
AMD64 Architecture Programmer's Manual, Volume 2	AMD64 Architecture Programmer's Manual, Volume 2: System Programming 24593 3.08	http://www.amd.com/us-en/Processors/DevelopWithAMD/
AMD64 Architecture Programmer's Manual, Volume 3	AMD64 Architecture Programmer's Manual, Volume 3: General Purpose and System Instructions 24594 3.03	http://www.amd.com/us-en/Processors/DevelopWithAMD/
AMD64 Architecture Programmer's Manual, Volume 4	AMD64 Architecture Programmer's Manual, Volume 4: 128-bit Media Instructions 26568 3.04	http://www.amd.com/us-en/Processors/DevelopWithAMD/
AMD64 Architecture Programmer's Manual, Volume 5	AMD64 Architecture Programmer's Manual, Volume 5: 64-bit Media and x87 Floating-Point Instructions 26569 3.03	http://www.amd.com/us-en/Processors/DevelopWithAMD/
DWARF Debugging Information Format	DWARF Debugging Information Format, Revision 2.0.0 (July 27, 1993)	http://www.eagercon.com/dwarf/dwarf-2.0.0.pdf
Filesystem Hierarchy Standard	Filesystem Hierarchy Standard (FHS) 2.3	http://www.pathname.com/fhs/
IEEE Std 754-1985	IEEE Standard 754 for Binary Floating-Point Arithmetic	http://www.ieee.org/
ISO C (1999)	ISO/IEC 9899: 1999, Programming Languages --C	
ISO POSIX (2003)	ISO/IEC 9945-1:2003 Information technology -- Portable Operating System Interface (POSIX) -- Part 1: Base Definitions ISO/IEC 9945-2:2003 Information technology -- Portable Operating	http://www.unix.org/version3/

Name	Title	URL
	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	
Large File Support	Large File Support	http://www.UNIX-systems.org/version2/whatsnew/lfs20mar.html
Li18nux Globalization Specification	LI18NUNIX 2000 Globalization Specification, Version 1.0 with Amendment 4	http://www.li18nux.org/docs/html/LI18NUNIX-2000-amd4.htm
Linux Allocated Device Registry	LINUX ALLOCATED DEVICES	http://www.lanana.org/docs/device-list/devices.txt
PAM	Open Software Foundation, Request For Comments: 86.0 , October 1995, V. Samar & R.Schemers (SunSoft)	http://www.opengroup.org/tech/rfc/mirror-rfc/rfc86.0.txt
RFC 1321: The MD5 Message-Digest Algorithm	IETF RFC 1321: The MD5 Message-Digest Algorithm	http://www.ietf.org/rfc/rfc1321.txt
RFC 1833: Binding Protocols for ONC RPC Version 2	IETF RFC 1833: Binding Protocols for ONC RPC Version 2	http://www.ietf.org/rfc/rfc1833.txt
RFC 1951: DEFLATE Compressed Data Format Specification	IETF RFC 1951: DEFLATE Compressed Data Format Specification version 1.3	http://www.ietf.org/rfc/rfc1951.txt
RFC 1952: GZIP File Format Specification	IETF RFC 1952: GZIP file format specification version 4.3	http://www.ietf.org/rfc/rfc1952.txt
RFC 2440: OpenPGP Message Format	IETF RFC 2440: OpenPGP Message Format	http://www.ietf.org/rfc/rfc2440.txt
SUSv2	CAE Specification, January 1997, System Interfaces and Headers (XSH), Issue 5 (ISBN: 1-85912-181-0, C606)	http://www.opengroup.org/publications/catalog/un.htm
SUSv2 Command and Utilities	The Single UNIX® Specification(SUS) Version 2, Commands and Utilities (XCU), Issue 5 (ISBN: 1-85912-191-8,	http://www.opengroup.org/publications/catalog/un.htm

Name	Title	URL
	C604)	
SVID Issue 3	American Telephone and Telegraph Company, System V Interface Definition, Issue 3 ; Morristown, NJ, UNIX Press, 1989.(ISBN 0201566524)	
SVID Issue 4	System V Interface Definition,Fourth Edition	
System V ABI	System V Application Binary Interface, Edition 4.1	http://www.caldera.com/developers/devspecs/gabi41.pdf
System V ABI Update	System V Application Binary Interface - DRAFT - 17 December 2003	http://www.caldera.com/developers/gabi/2003-12-17/contents.html
System V Application Binary Interface AMD64 Architecture Processor Supplement	System V Application Binary Interface AMD64 Architecture Processor Supplement, Draft Version 0.90	http://www.amd64.org/abi.pdf
this specification	Linux Standard Base	http://www.linuxbase.org/spec/
X/Open Curses	CAE Specification, May 1996, X/Open Curses, Issue 4, Version 2 (ISBN: 1-85912-171-3, C610), plus Corrigendum U018	http://www.opengroup.org/publications/catalog/un.htm
zlib Manual	zlib 1.2 Manual	http://www.gzip.org/zlib/

Chapter 3. Requirements

3.1. Relevant Libraries

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

5 **Table 3-1. Standard Library Names**

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

6
7 These libraries will be in an implementation-defined directory which the dynamic linker shall search by default.

3.2. LSB Implementation Conformance

8 A conforming implementation shall satisfy the following requirements:

- 9 • The implementation shall implement fully the architecture described in the hardware manual for the target
10 processor architecture.
- 11 • The implementation shall be capable of executing compiled applications having the format and using the system
12 interfaces described in this document.
- 13 • The implementation shall provide libraries containing the interfaces specified by this document, and shall provide a
14 dynamic linking mechanism that allows these interfaces to be attached to applications at runtime. All the interfaces
15 shall behave as specified in this document.
- 16 • The map of virtual memory provided by the implementation shall conform to the requirements of this document.
- 17 • The implementation's low-level behavior with respect to function call linkage, system traps, signals, and other such
18 activities shall conform to the formats described in this document.

- 19 • The implementation shall provide all of the mandatory interfaces in their entirety.
- 20 • The implementation may provide one or more of the optional interfaces. Each optional interface that is provided
- 21 shall be provided in its entirety. The product documentation shall state which optional interfaces are provided.
- 22 • The implementation shall provide all files and utilities specified as part of this document in the format defined here
- 23 and in other referenced documents. All commands and utilities shall behave as required by this document. The
- 24 implementation shall also provide all mandatory components of an application's runtime environment that are
- 25 included or referenced in this document.
- 26 • The implementation, when provided with standard data formats and values at a named interface, shall provide the
- 27 behavior defined for those values and data formats at that interface. However, a conforming implementation may
- 28 consist of components which are separately packaged and/or sold. For example, a vendor of a conforming
- 29 implementation might sell the hardware, operating system, and windowing system as separately packaged items.
- 30 • The implementation may provide additional interfaces with different names. It may also provide additional
- 31 behavior corresponding to data values outside the standard ranges, for standard named interfaces.

3.3. LSB Application Conformance

32 A conforming application shall satisfy the following requirements:

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

46 A strictly conforming application does not require or use any interface, facility, or implementation-defined extension

47 that is not defined in this document in order to be installed or to execute successfully.

Chapter 4. Definitions

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

3 can

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

5 cannot

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

7 may

8 is permitted; is allowed; is permissible

9 need not

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

11 shall

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

13 shall not

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

15 should

16 it is recommended that; ought to

17 should not

18 it is not recommended that; ought not to

Chapter 5. Terminology

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

2 archLSB

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

5 Binary Standard

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

7 gLSB

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

10 implementation-defined

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

16 Shell Script

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

19 Source Standard

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

21 undefined

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

27 unspecified

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

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

Chapter 6. Documentation Conventions

1 Throughout this document, the following typographic conventions are used:

2 `function()`

3 the name of a function

4 **command**

5 the name of a command or utility

6 `CONSTANT`

7 a constant value

8 *parameter*

9 a parameter

10 `variable`

11 a variable

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

14 `name`

15 the name of the interface

16 `(symver)`

17 An optional symbol version identifier, if required.

18 `[refno]`

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

20 For example,

21

<code>forkpty(GLIBC_2.0) [1]</code>

22 refers to the interface named `forkpty` with symbol version `GLIBC_2.0` that is defined in the first of the listed
23 references below the table.

ELF Specification

2

3 **ELF Specification**

Table of Contents

I. Low Level System Information	15
1. Machine Interface.....	1
1.1. Processor Architecture.....	1
1.2. Data Representation.....	1
1.2.1. Byte Ordering.....	1
1.2.2. Fundamental Types.....	1
1.2.3. Aggregates and Unions.....	1
1.2.4. Bit Fields.....	1
2. Function Calling Sequence.....	2
2.1. CPU Registers.....	2
2.2. Floating Point Registers.....	2
2.3. Stack Frame.....	2
2.4. Arguments.....	2
2.4.1. Integral/Pointer.....	2
2.4.2. Floating Point.....	2
2.4.3. Struct and Union Point.....	2
2.4.4. Variable Arguments.....	2
2.5. Return Values.....	2
2.5.1. Void.....	2
2.5.2. Integral/Pointer.....	2
2.5.3. Floating Point.....	2
2.5.4. Struct and Union Point.....	2
3. Operating System Interface.....	3
3.1. Virtual Address Space.....	3
3.1.1. Page Size.....	3
3.1.2. Virtual Address Assignments.....	3
3.1.3. Managing the PProcess Stack.....	3
3.1.4. Coding Guidelines.....	3
3.2. Processor Execution Mode.....	3
3.3. Exception Interface.....	3
3.3.1. Hardware Exception Types.....	3
3.3.2. Software Trap Types.....	3
3.4. Signal Delivery.....	3
3.4.1. Signal Handler Interface.....	3
4. Process Initialization.....	4
4.1. Special Registers.....	4
4.2. Process Stack (on entry).....	4
4.3. Auxilliary Vectors.....	4
4.4. Environment.....	4
5. Coding Examples.....	5
5.1. Code Model Overview/Architecture Constraints.....	5
5.2. Position-Independent Function Prologue.....	5
5.3. Data Objects.....	5

5.3.1. Absolute Load & Store.....	5
5.3.2. Position Relative Load & Store.....	5
5.4. Function Calls.....	5
5.4.1. Absolute Direct Function Call.....	5
5.4.2. Absolute Indirect Function Call.....	5
5.4.3. Position-Independent Direct Function Call.....	5
5.4.4. Position-Independent Indirect Function Call.....	5
5.5. Branching.....	5
5.5.1. Branch Instruction.....	5
5.5.2. Absolute switch() code.....	5
5.5.3. Position-Independent switch() code.....	5
6. C Stack Frame.....	6
6.1. Variable Argument List.....	6
6.2. Dynamic Allocation of Stack Space.....	6
7. Debug Information.....	7
II. Object Format.....	8
8. ELF Header.....	9
8.1. Machine Information.....	9
8.1.1. File Class.....	9
8.1.2. Data Encoding.....	9
8.1.3. OS Identification.....	9
8.1.4. Processor Identification.....	9
8.1.5. Processor Specific Flags.....	9
9. Sections.....	10
9.1. Special Sections.....	10
9.2. Additional Special Sections.....	10
10. Symbol Table.....	11
11. Relocation.....	12
11.1. Relocation Types.....	12
III. Program Loading and Dynamic Linking.....	13
12. Program Header.....	14
12.1. Types.....	14
12.2. Flags.....	14
13. Program Loading.....	15
14. Dynamic Linking.....	16
14.1. Dynamic Section.....	16
14.2. Global Offset Table.....	16
14.3. Shared Object Dependencies.....	16
14.4. Function Addresses.....	16
14.5. Procedure Linkage Table.....	16
14.6. Initialization and Termination Functions.....	16

List of Tables

9-1. ELF Special Sections.....	10
9-2. Additional Special Sections.....	10

I. Low Level System Information

Chapter 1. Machine Interface

1.1. Processor Architecture

- 1 The AMD64 Architecture is specified by the following documents
- 2 • AMD64 Architecture Programmer's Manual, Volume 1
 - 3 • AMD64 Architecture Programmer's Manual, Volume 2
 - 4 • AMD64 Architecture Programmer's Manual, Volume 3
 - 5 • AMD64 Architecture Programmer's Manual, Volume 4
 - 6 • AMD64 Architecture Programmer's Manual, Volume 5
 - 7 • System V Application Binary Interface AMD64 Architecture Processor Supplement
- 8 Applications conforming to this specification must provide feedback to the user if a feature that is required for correct
9 execution of the application is not present. Applications conforming to this specification should attempt to execute in
10 a diminished capacity if a required instruction set feature is not present.
- 11 Only instructions which do not require elevated privileges may be used.
- 12 Applications may not make system calls directly. The interfaces in the C library must be used instead.
- 13 This specification does not provide any performance guarantees of a conforming system. A system conforming to this
14 specification may be implemented in either hardware or software.

1.2. Data Representation

- 15 LSB-conforming applications shall use the data representation as defined in Chapter 3 of System V Application
16 Binary Interface AMD64 Architecture Processor Supplement.

1.2.1. Byte Ordering

1.2.2. Fundamental Types

1.2.3. Aggregates and Unions

1.2.4. Bit Fields

Chapter 2. Function Calling Sequence

- 1 LSB-conforming applications shall use the function calling sequence as defined in Chapter 3 of System V Application
- 2 Binary Interface AMD64 Architecture Processor Supplement.

2.1. CPU Registers

2.2. Floating Point Registers

2.3. Stack Frame

2.4. Arguments

2.4.1. Integral/Pointer

2.4.2. Floating Point

2.4.3. Struct and Union Point

2.4.4. Variable Arguments

2.5. Return Values

2.5.1. Void

2.5.2. Integral/Pointer

2.5.3. Floating Point

2.5.4. Struct and Union Point

Chapter 3. Operating System Interface

- 1 LSB-conforming applications shall use the Operating System Interfaces as defined in Chapter 3 of System V
- 2 Application Binary Interface AMD64 Architecture Processor Supplement.

3.1. Virtual Address Space

3.1.1. Page Size

3.1.2. Virtual Address Assignments

3.1.3. Managing the PProcess Stack

3.1.4. Coding Guidelines

3.2. Processor Execution Mode

3.3. Exception Interface

3.3.1. Hardware Exception Types

3.3.2. Software Trap Types

3.4. Signal Delivery

3.4.1. Signal Handler Interface

Chapter 4. Process Initialization

- 1 LSB-conforming applications shall use the Process Initialization as defined in Chapter 3 of the System V Application
- 2 Binary Interface AMD64 Architecture Processor Supplement.

4.1. Special Registers

4.2. Process Stack (on entry)

4.3. Auxilliary Vectors

4.4. Environment

Chapter 5. Coding Examples

- 1 LSB-conforming applications may implement fundamental operations using the Coding Examples as defined in
- 2 Chapter 3 of the System V Application Binary Interface AMD64 Architecture Processor Supplement.

5.1. Code Model Overview/Architecture Constraints

5.2. Position-Independent Function Prologue

5.3. Data Objects

5.3.1. Absolute Load & Store

5.3.2. Position Relative Load & Store

5.4. Function Calls

5.4.1. Absolute Direct Function Call

5.4.2. Absolute Indirect Function Call

5.4.3. Position-Independent Direct Function Call

5.4.4. Position-Independent Indirect Function Call

5.5. Branching

5.5.1. Branch Instruction

5.5.2. Absolute switch() code

5.5.3. Position-Independent switch() code

Chapter 6. C Stack Frame

6.1. Variable Argument List

6.2. Dynamic Allocation of Stack Space

Chapter 7. Debug Information

- 1 The LSB does not currently specify the format of Debug information.

II. Object Format

- 2 LSB-conforming implementations shall support an object file , called Executable and Linking Format (ELF) as
- 3 defined by the System V ABI , System V ABI Update , System V Application Binary Interface AMD64 Architecture
- 4 Processor Supplement and as supplemented by the this specification and this document.

Chapter 8. ELF Header

8.1. Machine Information

- 1 LSB-conforming applications shall use the Machine Information as defined in Chapter 4 of the System V Application
- 2 Binary Interface AMD64 Architecture Processor Supplement.

8.1.1. File Class

8.1.2. Data Encoding

8.1.3. OS Identification

8.1.4. Processor Identification

8.1.5. Processor Specific Flags

Chapter 9. Sections

9.1. Special Sections

1 The following sections are defined in the System V Application Binary Interface AMD64 Architecture Processor
2 Supplement.

3 **Table 9-1. ELF Special Sections**

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

5 .got

6 This section holds the global offset table

7 .plt

8 This section holds the procedure linkage table.

9.2. Additional Special Sections

9 The following additional sections are defined here.

10 **Table 9-2. Additional Special Sections**

Name	Type	Attributes
.rela.dyn	SHT_RELA	SHF_ALLOC
.rela.plt	SHT_RELA	SHF_ALLOC

12 .rela.dyn

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

14 .rela.plt

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

Chapter 10. Symbol Table

- 1 LSB-conforming applications shall use the Symbol Table as defined in Chapter 4 of the System V Application Binary
- 2 Interface AMD64 Architecture Processor Supplement.

Chapter 11. Relocation

- 1 LSB-conforming applications shall use Relocations as defined in Chapter 4 of the System V Application Binary
- 2 Interface AMD64 Architecture Processor Supplement.

11.1. Relocation Types

III. Program Loading and Dynamic Linking

- 2 LSB-conforming implementations shall support the object file information and system actions that create running
- 3 programs as specified in the System V ABI , System V ABI Update , System V Application Binary Interface AMD64
- 4 Architecture Processor Supplement and as supplemented by the this specification and this document.

Chapter 12. Program Header

12.1. Types

12.2. Flags

Chapter 13. Program Loading

Chapter 14. Dynamic Linking

14.1. Dynamic Section

1 The following dynamic entries are defined in the System V Application Binary Interface AMD64 Architecture
2 Processor Supplement.

3 DT_JMPREL

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

6 DT_PLTGOT

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

8 DT_RELACOUNT

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

14.2. Global Offset Table

10 The Global Offset Table is defined in Chapter 5 of System V Application Binary Interface AMD64 Architecture
11 Processor Supplement.

14.3. Shared Object Dependencies

14.4. Function Addresses

14.5. Procedure Linkage Table

14.6. Initialization and Termination Functions

Linux Standard Base Specification

2

3 **Linux Standard Base Specification**

Table of Contents

I. Base Libraries	23
1. Libraries	1
1.1. Program Interpreter/Dynamic Linker	1
1.2. Interfaces for libc	1
1.2.1. RPC	1
1.2.1.1. Interfaces for RPC	1
1.2.2. System Calls	2
1.2.2.1. Interfaces for System Calls	2
1.2.3. Standard I/O	4
1.2.3.1. Interfaces for Standard I/O	4
1.2.4. Signal Handling	5
1.2.4.1. Interfaces for Signal Handling	5
1.2.5. Localization Functions	6
1.2.5.1. Interfaces for Localization Functions	6
1.2.6. Socket Interface	7
1.2.6.1. Interfaces for Socket Interface	7
1.2.7. Wide Characters	8
1.2.7.1. Interfaces for Wide Characters	8
1.2.8. String Functions	9
1.2.8.1. Interfaces for String Functions	9
1.2.9. IPC Functions	10
1.2.9.1. Interfaces for IPC Functions	10
1.2.10. Regular Expressions	11
1.2.10.1. Interfaces for Regular Expressions	11
1.2.11. Character Type Functions	11
1.2.11.1. Interfaces for Character Type Functions	11
1.2.12. Time Manipulation	12
1.2.12.1. Interfaces for Time Manipulation	12
1.2.13. Terminal Interface Functions	13
1.2.13.1. Interfaces for Terminal Interface Functions	13
1.2.14. System Database Interface	14
1.2.14.1. Interfaces for System Database Interface	14
1.2.15. Language Support	14
1.2.15.1. Interfaces for Language Support	14
1.2.16. Large File Support	15
1.2.16.1. Interfaces for Large File Support	15
1.2.17. Standard Library	15
1.2.17.1. Interfaces for Standard Library	15
1.3. Data Definitions for libc	17
1.3.1. errno.h	18
1.3.2. inttypes.h	18
1.3.3. limits.h	18
1.3.4. setjmp.h	18

1.3.5. signal.h	18
1.3.6. stddef.h	20
1.3.7. sys/ioctl.h	20
1.3.8. sys/ipc.h	20
1.3.9. sys/mman.h	20
1.3.10. sys/msg.h	20
1.3.11. sys/sem.h	21
1.3.12. sys/shm.h	21
1.3.13. sys/socket.h	22
1.3.14. sys/stat.h	22
1.3.15. sys/statvfs.h	23
1.3.16. sys/types.h	23
1.3.17. termios.h	23
1.3.18. ucontext.h	25
1.3.19. unistd.h	26
1.3.20. utmp.h	26
1.3.21. utmpx.h	26
1.4. Interfaces for libm	27
1.4.1. Math	27
1.4.1.1. Interfaces for Math	27
1.5. Interfaces for libpthread	31
1.5.1. Realtime Threads	31
1.5.1.1. Interfaces for Realtime Threads	31
1.5.2. Advanced Realtime Threads	31
1.5.2.1. Interfaces for Advanced Realtime Threads	31
1.5.3. Posix Threads	31
1.5.3.1. Interfaces for Posix Threads	31
1.6. Interfaces for libgcc_s	33
1.6.1. Unwind Library	33
1.6.1.1. Interfaces for Unwind Library	33
1.7. Interface Definitions for libgcc_s	34
_Unwind_DeleteException	34
_Unwind_Find_FDE	34
_Unwind_ForcedUnwind	35
_Unwind_GetDataRelBase	36
_Unwind_GetGR	36
_Unwind_GetIP	36
_Unwind_GetLanguageSpecificData	37
_Unwind_GetRegionStart	37
_Unwind_GetTextRelBase	37
_Unwind_RaiseException	38
_Unwind_Resume	39
_Unwind_SetGR	39
_Unwind_SetIP	39
1.8. Interfaces for libdl	39
1.8.1. Dynamic Loader	40
1.8.1.1. Interfaces for Dynamic Loader	40
1.9. Interfaces for libcrypt	40

1.9.1. Encryption	40
1.9.1.1. Interfaces for Encryption	40
II. Utility Libraries	42
2. Libraries	43
2.1. Interfaces for libz	43
2.1.1. Compression Library	43
2.1.1.1. Interfaces for Compression Library	43
2.2. Interfaces for libncurses	43
2.2.1. Curses	43
2.2.1.1. Interfaces for Curses	43
2.3. Interfaces for libutil	43
2.3.1. Utility Functions	44
2.3.1.1. Interfaces for Utility Functions	44
A. Alphabetical Listing of Interfaces	45
A.1. libgcc_s	45

List of Tables

1-1. libc Definition.....	1
1-2. libc - RPC Function Interfaces	1
1-3. libc - System Calls Function Interfaces	2
1-4. libc - Standard I/O Function Interfaces	4
1-5. libc - Standard I/O Data Interfaces	5
1-6. libc - Signal Handling Function Interfaces	5
1-7. libc - Signal Handling Data Interfaces.....	6
1-8. libc - Localization Functions Function Interfaces	6
1-9. libc - Localization Functions Data Interfaces	7
1-10. libc - Socket Interface Function Interfaces	7
1-11. libc - Socket Interface Deprecated Function Interfaces	8
1-12. libc - Wide Characters Function Interfaces	8
1-13. libc - String Functions Function Interfaces.....	9
1-14. libc - IPC Functions Function Interfaces	10
1-15. libc - Regular Expressions Function Interfaces	11
1-16. libc - Regular Expressions Deprecated Function Interfaces	11
1-17. libc - Regular Expressions Deprecated Data Interfaces.....	11
1-18. libc - Character Type Functions Function Interfaces.....	12
1-19. libc - Time Manipulation Function Interfaces	12
1-20. libc - Time Manipulation Deprecated Function Interfaces	13
1-21. libc - Time Manipulation Data Interfaces.....	13
1-22. libc - Terminal Interface Functions Function Interfaces.....	13
1-23. libc - System Database Interface Function Interfaces.....	14
1-24. libc - Language Support Function Interfaces.....	14
1-25. libc - Large File Support Function Interfaces	15
1-26. libc - Standard Library Function Interfaces	15
1-27. libc - Standard Library Data Interfaces	17
1-28. libm Definition	27
1-29. libm - Math Function Interfaces	27
1-30. libm - Math Data Interfaces.....	31
1-31. libpthread Definition	31
1-32. libpthread - Posix Threads Function Interfaces	31
1-33. libgcc_s Definition	33
1-34. libgcc_s - Unwind Library Function Interfaces	33
1-35. libdl Definition	40
1-36. libdl - Dynamic Loader Function Interfaces	40
1-37. libcrypt Definition	40
1-38. libcrypt - Encryption Function Interfaces	40
2-1. libz Definition.....	43
2-2. libncurses Definition	43
2-3. libutil Definition	43
2-4. libutil - Utility Functions Function Interfaces	44
A-1. libgcc_s Function Interfaces	45

I. Base Libraries

Chapter 1. Libraries

- 1 An LSB-conforming implementation shall support some base libraries which provide interfaces for accessing the
2 operating system, processor and other hardware in the system.
- 3 Interfaces that are unique to the AMD64 platform are defined here. This section should be used in conjunction with the
4 corresponding section in the Linux Standard Base Specification.

1.1. Program Interpreter/Dynamic Linker

- 5 The LSB specifies the Program Interpreter to be `/lib64/ld-lsb-x86-64.so.2`.

1.2. Interfaces for libc

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

7 **Table 1-1. libc Definition**

Library:	libc
SONAME:	libc.so.6

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

Large File Support
this specification
SUSv2
ISO POSIX (2003)
SVID Issue 3
SVID Issue 4

10

1.2.1. RPC

1.2.1.1. Interfaces for RPC

- 11 An LSB conforming implementation shall provide the architecture specific functions for RPC specified in Table 1-2,
12 with the full functionality as described in the referenced underlying specification.

14 **Table 1-2. libc - RPC Function Interfaces**

<code>authnone_create(GLIBC_2.2.5)</code> [1]	<code>pmap_unset(GLIBC_2.2.5)</code> [2]	<code>svcerr_weakauth(GLIBC_2.2.5)</code> [3]	<code>xdr_float(GLIBC_2.2.5)</code> [3]	<code>xdr_u_char(GLIBC_2.2.5)</code> [3]
<code>clnt_create(GLIBC_2.2.5)</code> [1]	<code>setdomainname(GLIBC_2.2.5)</code> [2]	<code>svctcp_create(GLIBC_2.2.5)</code> [2]	<code>xdr_free(GLIBC_2.2.5)</code> [3]	<code>xdr_u_int(GLIBC_2.2.5)</code> [2]
<code>clnt_pcreateerror(GLIBC_2.2.5)</code> [1]	<code>svc_getreqset(GLIBC_2.2.5)</code> [3]	<code>svcudp_create(GLIBC_2.2.5)</code> [2]	<code>xdr_int(GLIBC_2.2.5)</code> [3]	<code>xdr_u_long(GLIBC_2.2.5)</code> [3]

clnt_perrno(GLIBC_2.2.5) [1]	svc_register(GLIBC_2.2.5) [2]	xdr_accepted_reply(GLIBC_2.2.5) [3]	xdr_long(GLIBC_2.2.5) [3]	xdr_u_short(GLIBC_2.2.5) [3]
clnt_perror(GLIBC_2.2.5) [1]	svc_run(GLIBC_2.2.5) [2]	xdr_array(GLIBC_2.2.5) [3]	xdr_opaque(GLIBC_2.2.5) [3]	xdr_union(GLIBC_2.2.5) [3]
clnt_spcreateerror(GLIBC_2.2.5) [1]	svc_sendreply(GLIBC_2.2.5) [2]	xdr_bool(GLIBC_2.2.5) [3]	xdr_opaque_auth(GLIBC_2.2.5) [3]	xdr_vector(GLIBC_2.2.5) [3]
clnt_sperrno(GLIBC_2.2.5) [1]	svcerr_auth(GLIBC_2.2.5) [3]	xdr_bytes(GLIBC_2.2.5) [3]	xdr_pointer(GLIBC_2.2.5) [3]	xdr_void(GLIBC_2.2.5) [3]
clnt_sperror(GLIBC_2.2.5) [1]	svcerr_decode(GLIBC_2.2.5) [3]	xdr_callhdr(GLIBC_2.2.5) [3]	xdr_reference(GLIBC_2.2.5) [3]	xdr_wrapstring(GLIBC_2.2.5) [3]
getdomainname(GLIBC_2.2.5) [2]	svcerr_noproc(GLIBC_2.2.5) [3]	xdr_callmsg(GLIBC_2.2.5) [3]	xdr_rejected_reply(GLIBC_2.2.5) [3]	xdrmem_create(GLIBC_2.2.5) [3]
key_decryptsession(GLIBC_2.2.5) [3]	svcerr_noprog(GLIBC_2.2.5) [3]	xdr_char(GLIBC_2.2.5) [3]	xdr_replymsg(GLIBC_2.2.5) [3]	xdrrec_create(GLIBC_2.2.5) [3]
pmap_getport(GLIBC_2.2.5) [2]	svcerr_progvers(GLIBC_2.2.5) [3]	xdr_double(GLIBC_2.2.5) [3]	xdr_short(GLIBC_2.2.5) [3]	xdrrec_eof(GLIBC_2.2.5) [3]
pmap_set(GLIBC_2.2.5) [2]	svcerr_systemerr(GLIBC_2.2.5) [3]	xdr_enum(GLIBC_2.2.5) [3]	xdr_string(GLIBC_2.2.5) [3]	

15

16 *Referenced Specification(s)*

17 [1]. SVID Issue 4

18 [2]. this specification

19 [3]. SVID Issue 3

1.2.2. System Calls

1.2.2.1. Interfaces for System Calls

21 An LSB conforming implementation shall provide the architecture specific functions for System Calls specified in
22 Table 1-3, with the full functionality as described in the referenced underlying specification.

23 **Table 1-3. libc - System Calls Function Interfaces**

__fxstat(GLIBC_2.2.5) [1]	fchmod(GLIBC_2.2.5) [2]	getwd(GLIBC_2.2.5) [2]	read(GLIBC_2.2.5) [2]	setrlimit(GLIBC_2.2.5) [2]
__getpgid(GLIBC_2.2.5) [1]	fchown(GLIBC_2.2.5) [2]	initgroups(GLIBC_2.2.5) [1]	readdir(GLIBC_2.2.5) [2]	setrlimit64(GLIBC_2.2.5) [3]
__lxstat(GLIBC_2.2.5) [1]	fcntl(GLIBC_2.2.5) [1]	ioctl(GLIBC_2.2.5) [1]	readdir_r(GLIBC_2.2.5) [2]	setsid(GLIBC_2.2.5) [2]
__xmknod(GLIBC_2.2.5) [1]	fdatasync(GLIBC_2.2.5) [2]	kill(GLIBC_2.2.5) [1]	readlink(GLIBC_2.2.5) [2]	setuid(GLIBC_2.2.5) [2]

<code>__xstat(GLIBC_2.2.5)</code> [1]	<code>flock(GLIBC_2.2.5)</code> [1]	<code>killpg(GLIBC_2.2.5)</code> [2]	<code>readv(GLIBC_2.2.5)</code> [2]	<code>sleep(GLIBC_2.2.5)</code> [2]
<code>access(GLIBC_2.2.5)</code> [2]	<code>fork(GLIBC_2.2.5)</code> [2]	<code>lchown(GLIBC_2.2.5)</code> [2]	<code>rename(GLIBC_2.2.5)</code> [2]	<code>statvfs(GLIBC_2.2.5)</code> [2]
<code>acct(GLIBC_2.2.5)</code> [1]	<code>fstatvfs(GLIBC_2.2.5)</code> [2]	<code>link(GLIBC_2.2.5)</code> [2]	<code>rmdir(GLIBC_2.2.5)</code> [2]	<code>stime(GLIBC_2.2.5)</code> [1]
<code>alarm(GLIBC_2.2.5)</code> [2]	<code>fsync(GLIBC_2.2.5)</code> [2]	<code>lockf(GLIBC_2.2.5)</code> [2]	<code>sbrk(GLIBC_2.2.5)</code> [4]	<code>symlink(GLIBC_2.2.5)</code> [2]
<code>brk(GLIBC_2.2.5)</code> [4]	<code>ftime(GLIBC_2.2.5)</code> [2]	<code>lseek(GLIBC_2.2.5)</code> [2]	<code>sched_get_priority_max(GLIBC_2.2.5)</code> [2]	<code>sync(GLIBC_2.2.5)</code> [2]
<code>chdir(GLIBC_2.2.5)</code> [2]	<code>ftruncate(GLIBC_2.2.5)</code> [2]	<code>mkdir(GLIBC_2.2.5)</code> [2]	<code>sched_get_priority_min(GLIBC_2.2.5)</code> [2]	<code>sysconf(GLIBC_2.2.5)</code> [2]
<code>chmod(GLIBC_2.2.5)</code> [2]	<code>getcontext(GLIBC_2.2.5)</code> [2]	<code>mkfifo(GLIBC_2.2.5)</code> [2]	<code>sched_getparam(GLIBC_2.2.5)</code> [2]	<code>time(GLIBC_2.2.5)</code> [2]
<code>chown(GLIBC_2.2.5)</code> [2]	<code>getegid(GLIBC_2.2.5)</code> [2]	<code>mlock(GLIBC_2.2.5)</code> [2]	<code>sched_getscheduler(GLIBC_2.2.5)</code> [2]	<code>times(GLIBC_2.2.5)</code> [2]
<code>chroot(GLIBC_2.2.5)</code> [4]	<code>geteuid(GLIBC_2.2.5)</code> [2]	<code>mlockall(GLIBC_2.2.5)</code> [2]	<code>sched_rr_get_interval(GLIBC_2.2.5)</code> [2]	<code>truncate(GLIBC_2.2.5)</code> [2]
<code>clock(GLIBC_2.2.5)</code> [2]	<code>getgid(GLIBC_2.2.5)</code> [2]	<code>mmap(GLIBC_2.2.5)</code> [2]	<code>sched_setparam(GLIBC_2.2.5)</code> [2]	<code>ulimit(GLIBC_2.2.5)</code> [2]
<code>close(GLIBC_2.2.5)</code> [2]	<code>getgroups(GLIBC_2.2.5)</code> [2]	<code>mprotect(GLIBC_2.2.5)</code> [2]	<code>sched_setscheduler(GLIBC_2.2.5)</code> [2]	<code>umask(GLIBC_2.2.5)</code> [2]
<code>closedir(GLIBC_2.2.5)</code> [2]	<code>getitimer(GLIBC_2.2.5)</code> [2]	<code>msync(GLIBC_2.2.5)</code> [2]	<code>sched_yield(GLIBC_2.2.5)</code> [2]	<code>uname(GLIBC_2.2.5)</code> [2]
<code>creat(GLIBC_2.2.5)</code> [1]	<code>getloadavg(GLIBC_2.2.5)</code> [1]	<code>munlock(GLIBC_2.2.5)</code> [2]	<code>select(GLIBC_2.2.5)</code> [2]	<code>unlink(GLIBC_2.2.5)</code> [1]
<code>dup(GLIBC_2.2.5)</code> [2]	<code>getpagesize(GLIBC_2.2.5)</code> [4]	<code>munlockall(GLIBC_2.2.5)</code> [2]	<code>setcontext(GLIBC_2.2.5)</code> [2]	<code>utime(GLIBC_2.2.5)</code> [2]
<code>dup2(GLIBC_2.2.5)</code> [2]	<code>getpgid(GLIBC_2.2.5)</code> [2]	<code>munmap(GLIBC_2.2.5)</code> [2]	<code>setegid(GLIBC_2.2.5)</code> [2]	<code>utimes(GLIBC_2.2.5)</code> [2]
<code>execl(GLIBC_2.2.5)</code> [2]	<code>getpgrp(GLIBC_2.2.5)</code> [2]	<code>nanosleep(GLIBC_2.2.5)</code> [2]	<code>seteuid(GLIBC_2.2.5)</code> [2]	<code>vfork(GLIBC_2.2.5)</code> [2]
<code>execle(GLIBC_2.2.5)</code> [2]	<code>getpid(GLIBC_2.2.5)</code> [2]	<code>nice(GLIBC_2.2.5)</code> [2]	<code>setgid(GLIBC_2.2.5)</code> [2]	<code>wait(GLIBC_2.2.5)</code> [2]
<code>execlp(GLIBC_2.2.5)</code> [2]	<code>getppid(GLIBC_2.2.5)</code> [2]	<code>open(GLIBC_2.2.5)</code> [1]	<code>setitimer(GLIBC_2.2.5)</code> [2]	<code>wait3(GLIBC_2.2.5)</code> [1]

execv(GLIBC_2.2.5) [2]	getpriority(GLIBC_2.2.5) [2]	opendir(GLIBC_2.2.5) [2]	setpgid(GLIBC_2.2.5) [2]	wait4(GLIBC_2.2.5) [1]
execve(GLIBC_2.2.5) [2]	getrlimit(GLIBC_2.2.5) [2]	pathconf(GLIBC_2.2.5) [2]	setpgrp(GLIBC_2.2.5) [2]	waitpid(GLIBC_2.2.5) [1]
execvp(GLIBC_2.2.5) [2]	getrusage(GLIBC_2.2.5) [2]	pause(GLIBC_2.2.5) [2]	setpriority(GLIBC_2.2.5) [2]	write(GLIBC_2.2.5) [2]
exit(GLIBC_2.2.5) [2]	getsid(GLIBC_2.2.5) [2]	pipe(GLIBC_2.2.5) [2]	setregid(GLIBC_2.2.5) [2]	writew(GLIBC_2.2.5) [2]
fchdir(GLIBC_2.2.5) [2]	getuid(GLIBC_2.2.5) [2]	poll(GLIBC_2.2.5) [2]	setreuid(GLIBC_2.2.5) [2]	

24

25 *Referenced Specification(s)*

26 [1]. this specification

27 [2]. ISO POSIX (2003)

28 [3]. Large File Support

29 [4]. SUSv2

1.2.3. Standard I/O

1.2.3.1. Interfaces for Standard I/O

31 An LSB conforming implementation shall provide the architecture specific functions for Standard I/O specified in
32 Table 1-4, with the full functionality as described in the referenced underlying specification.

33 **Table 1-4. libc - Standard I/O Function Interfaces**

_IO_feof(GLIBC_2.2.5) [1]	fgetpos(GLIBC_2.2.5) [2]	fsetpos(GLIBC_2.2.5) [2]	putchar(GLIBC_2.2.5) [2]	sscanf(GLIBC_2.2.5) [2]
_IO_getc(GLIBC_2.2.5) [1]	fgets(GLIBC_2.2.5) [2]	ftell(GLIBC_2.2.5) [2]	putchar_unlocked(GLIBC_2.2.5) [2]	telldir(GLIBC_2.2.5) [2]
_IO_putc(GLIBC_2.2.5) [1]	fgetwc_unlocked(GLIBC_2.2.5) [1]	ftello(GLIBC_2.2.5) [2]	puts(GLIBC_2.2.5) [2]	tempnam(GLIBC_2.2.5) [2]
_IO_puts(GLIBC_2.2.5) [1]	fileno(GLIBC_2.2.5) [2]	fwrite(GLIBC_2.2.5) [2]	putw(GLIBC_2.2.5) [3]	ungetc(GLIBC_2.2.5) [2]
asprintf(GLIBC_2.2.5) [1]	flockfile(GLIBC_2.2.5) [2]	getc(GLIBC_2.2.5) [2]	remove(GLIBC_2.2.5) [2]	vasprintf(GLIBC_2.2.5) [1]
clearerr(GLIBC_2.2.5) [2]	fopen(GLIBC_2.2.5) [1]	getc_unlocked(GLIBC_2.2.5) [2]	rewind(GLIBC_2.2.5) [2]	vdprintf(GLIBC_2.2.5) [1]
ctermid(GLIBC_2.2.5) [2]	fprintf(GLIBC_2.2.5) [2]	getchar(GLIBC_2.2.5) [2]	rewinddir(GLIBC_2.2.5) [2]	vfprintf(GLIBC_2.2.5) [2]

fclose(GLIBC_2.2.5) [2]	fputc(GLIBC_2.2.5) [2]	getchar_unlocked(GLIBC_2.2.5) [2]	scanf(GLIBC_2.2.5) [2]	vprintf(GLIBC_2.2.5) [2]
fdopen(GLIBC_2.2.5) [2]	fputs(GLIBC_2.2.5) [2]	getw(GLIBC_2.2.5) [3]	seekdir(GLIBC_2.2.5) [2]	vsprintf(GLIBC_2.2.5) [2]
feof(GLIBC_2.2.5) [2]	fread(GLIBC_2.2.5) [2]	pclose(GLIBC_2.2.5) [2]	setbuf(GLIBC_2.2.5) [2]	vsprintf(GLIBC_2.2.5) [2]
ferror(GLIBC_2.2.5) [2]	freopen(GLIBC_2.2.5) [1]	popen(GLIBC_2.2.5) [2]	setbuffer(GLIBC_2.2.5) [1]	
fflush(GLIBC_2.2.5) [2]	fscanf(GLIBC_2.2.5) [2]	printf(GLIBC_2.2.5) [2]	setvbuf(GLIBC_2.2.5) [2]	
fflush_unlocked(GLIBC_2.2.5) [1]	fseek(GLIBC_2.2.5) [2]	putc(GLIBC_2.2.5) [2]	snprintf(GLIBC_2.2.5) [2]	
fgetc(GLIBC_2.2.5) [2]	fseeko(GLIBC_2.2.5) [2]	putc_unlocked(GLIBC_2.2.5) [2]	sprintf(GLIBC_2.2.5) [2]	

34

35 *Referenced Specification(s)*

36 [1]. this specification

37 [2]. ISO POSIX (2003)

38 [3]. SUSv2

39 An LSB conforming implementation shall provide the architecture specific data interfaces for Standard I/O specified
 40 in Table 1-5, with the full functionality as described in the referenced underlying specification.

41 **Table 1-5. libc - Standard I/O Data Interfaces**

stderr(GLIBC_2.2.5) [1]	stdin(GLIBC_2.2.5) [1]	stdout(GLIBC_2.2.5) [1]		
-------------------------	------------------------	-------------------------	--	--

42

43 *Referenced Specification(s)*

44 [1]. ISO POSIX (2003)

1.2.4. Signal Handling

1.2.4.1. Interfaces for Signal Handling

46 An LSB conforming implementation shall provide the architecture specific functions for Signal Handling specified in
 47 Table 1-6, with the full functionality as described in the referenced underlying specification.

48 **Table 1-6. libc - Signal Handling Function Interfaces**

__libc_current_sigrtmax(GLIBC_2.2.5) [1]	sigaddset(GLIBC_2.2.5) [2]	sighold(GLIBC_2.2.5) [2]	sigpause(GLIBC_2.2.5) [2]	sigsuspend(GLIBC_2.2.5) [2]
__libc_current_sigrt	sigaltstack(GLIBC_2.2.5) [2]	sigignore(GLIBC_2.2.5) [2]	sigpending(GLIBC_2.2.5) [2]	sigtimedwait(GLIBC_2.2.5) [2]

min(GLIBC_2.2.5) [1]	2.2.5) [2]	.2.5) [2]	2.2.5) [2]	C_2.2.5) [2]
__sigsetjmp(GLIBC_2.2.5) [1]	sigandset(GLIBC_2.2.5) [1]	siginterrupt(GLIBC_2.2.5) [2]	sigprocmask(GLIBC_2.2.5) [2]	sigwait(GLIBC_2.2.5) [2]
__sysv_signal(GLIBC_2.2.5) [1]	sigblock(GLIBC_2.2.5) [1]	sigisemptyset(GLIBC_2.2.5) [1]	sigqueue(GLIBC_2.2.5) [2]	sigwaitinfo(GLIBC_2.2.5) [2]
bsd_signal(GLIBC_2.2.5) [2]	sigdelset(GLIBC_2.2.5) [2]	sigismember(GLIBC_2.2.5) [2]	sigrelse(GLIBC_2.2.5) [2]	
psignal(GLIBC_2.2.5) [1]	sigemptyset(GLIBC_2.2.5) [2]	siglongjmp(GLIBC_2.2.5) [2]	sigreturn(GLIBC_2.2.5) [1]	
raise(GLIBC_2.2.5) [2]	sigfillset(GLIBC_2.2.5) [2]	signal(GLIBC_2.2.5) [2]	sigset(GLIBC_2.2.5) [2]	
sigaction(GLIBC_2.2.5) [2]	siggetmask(GLIBC_2.2.5) [1]	sigorset(GLIBC_2.2.5) [1]	sigstack(GLIBC_2.2.5) [3]	

49

50 *Referenced Specification(s)*

51 [1]. this specification

52 [2]. ISO POSIX (2003)

53 [3]. SUSv2

54 An LSB conforming implementation shall provide the architecture specific data interfaces for Signal Handling
 55 specified in Table 1-7, with the full functionality as described in the referenced underlying specification.

56 **Table 1-7. libc - Signal Handling Data Interfaces**

__sys_siglist(GLIBC_2.3.3) [1]				
--------------------------------	--	--	--	--

57

58 *Referenced Specification(s)*

59 [1]. this specification

1.2.5. Localization Functions

1.2.5.1. Interfaces for Localization Functions

61 An LSB conforming implementation shall provide the architecture specific functions for Localization Functions
 62 specified in Table 1-8, with the full functionality as described in the referenced underlying specification.

63 **Table 1-8. libc - Localization Functions Function Interfaces**

bind_textdomain_codeset(GLIBC_2.2.5) [1]	catopen(GLIBC_2.2.5) [2]	dngettext(GLIBC_2.2.5) [1]	iconv_open(GLIBC_2.2.5) [2]	setlocale(GLIBC_2.2.5) [2]
bindtextdomain(GLIBC_2.2.5) [1]	dcgettext(GLIBC_2.2.5) [1]	gettext(GLIBC_2.2.5) [1]	localeconv(GLIBC_2.2.5) [1]	textdomain(GLIBC_2.2.5) [1]

IBC_2.2.5) [1]	2.5) [1]	5) [1]	2.2.5) [2]	_2.2.5) [1]
catclose(GLIBC_2.2.5) [2]	dcngettext(GLIBC_2.2.5) [1]	iconv(GLIBC_2.2.5) [2]	ngettext(GLIBC_2.2.5) [1]	
catgets(GLIBC_2.2.5) [2]	dgettext(GLIBC_2.2.5) [1]	iconv_close(GLIBC_2.2.5) [2]	nl_langinfo(GLIBC_2.2.5) [2]	

64

65 *Referenced Specification(s)*

66 [1]. this specification

67 [2]. ISO POSIX (2003)

68 An LSB conforming implementation shall provide the architecture specific data interfaces for Localization Functions
 69 specified in Table 1-9, with the full functionality as described in the referenced underlying specification.

70 **Table 1-9. libc - Localization Functions Data Interfaces**

_nl_msg_cat_cntr(GLIBC_2.2.5) [1]				
-----------------------------------	--	--	--	--

71

72 *Referenced Specification(s)*

73 [1]. this specification

1.2.6. Socket Interface

1.2.6.1. Interfaces for Socket Interface

74 An LSB conforming implementation shall provide the architecture specific functions for Socket Interface specified in
 75 Table 1-10, with the full functionality as described in the referenced underlying specification.

77 **Table 1-10. libc - Socket Interface Function Interfaces**

__h_errno_location(GLIBC_2.2.5) [1]	gethostid(GLIBC_2.2.5) [2]	listen(GLIBC_2.2.5) [2]	sendmsg(GLIBC_2.2.5) [2]	socketpair(GLIBC_2.2.5) [2]
accept(GLIBC_2.2.5) [2]	gethostname(GLIBC_2.2.5) [2]	recv(GLIBC_2.2.5) [2]	sendto(GLIBC_2.2.5) [2]	
bind(GLIBC_2.2.5) [2]	getpeername(GLIBC_2.2.5) [2]	recvfrom(GLIBC_2.2.5) [2]	setsockopt(GLIBC_2.2.5) [1]	
bindresvport(GLIBC_2.2.5) [1]	getsockname(GLIBC_2.2.5) [2]	recvmsg(GLIBC_2.2.5) [2]	shutdown(GLIBC_2.2.5) [2]	
connect(GLIBC_2.2.5) [2]	getsockopt(GLIBC_2.2.5) [2]	send(GLIBC_2.2.5) [2]	socket(GLIBC_2.2.5) [2]	

78

79 *Referenced Specification(s)*

80 [1]. this specification

81 [2]. ISO POSIX (2003)

82 An LSB conforming implementation shall provide the architecture specific deprecated functions for Socket Interface
83 specified in Table 1-11, with the full functionality as described in the referenced underlying specification.

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

86 **Table 1-11. libc - Socket Interface Deprecated Function Interfaces**

gethostbyname_r(G LIBC_2.2.5) [1]				
--------------------------------------	--	--	--	--

87
88 *Referenced Specification(s)*

89 [1]. this specification

1.2.7. Wide Characters

1.2.7.1. Interfaces for Wide Characters

90
91 An LSB conforming implementation shall provide the architecture specific functions for Wide Characters specified in
92 Table 1-12, with the full functionality as described in the referenced underlying specification.

93 **Table 1-12. libc - Wide Characters Function Interfaces**

__wctod_internal(GLIBC_2.2.5) [1]	mbsinit(GLIBC_2.2. .5) [2]	vwscanf(GLIBC_2. 2.5) [2]	wcsnlen(GLIBC_2. 2.5) [1]	wcstoumax(GLIBC _2.2.5) [2]
__wctof_internal(GLIBC_2.2.5) [1]	mbsnrto wcs(GLIBC _2.2.5) [1]	wcpcpy(GLIBC_2.2 .5) [1]	wcsnrto mbs(GLIBC _2.2.5) [1]	wcstouq(GLIBC_2. 2.5) [1]
__wctol_internal(G LIBC_2.2.5) [1]	mbsrtowcs(GLIBC_ 2.2.5) [2]	wcpncpy(GLIBC_2. 2.5) [1]	wcsprk(GLIBC_2. 2.5) [2]	wcswcs(GLIBC_2.2 .5) [2]
__wctold_internal(GLIBC_2.2.5) [1]	mbstowcs(GLIBC_ 2.2.5) [2]	wcrtomb(GLIBC_2. 2.5) [2]	wcsrchr(GLIBC_2.2 .5) [2]	wcswidth(GLIBC_2 .2.5) [2]
__wctoul_internal(GLIBC_2.2.5) [1]	mbtowc(GLIBC_2. 2.5) [2]	wcscasecmp(GLIB C_2.2.5) [1]	wcsrtombs(GLIBC_ 2.2.5) [2]	wcsxfrm(GLIBC_2. 2.5) [2]
btowc(GLIBC_2.2. 5) [2]	putwc(GLIBC_2.2. 5) [2]	wcscat(GLIBC_2.2. 5) [2]	wcsspn(GLIBC_2.2 .5) [2]	wctob(GLIBC_2.2. 5) [2]
fgetwc(GLIBC_2.2. 5) [2]	putwchar(GLIBC_2 .2.5) [2]	wcschr(GLIBC_2.2. 5) [2]	wcsstr(GLIBC_2.2. 5) [2]	wctomb(GLIBC_2. 2.5) [2]
fgetws(GLIBC_2.2. 5) [2]	swprintf(GLIBC_2. 2.5) [2]	wcscmp(GLIBC_2. 2.5) [2]	wctod(GLIBC_2.2. 5) [2]	wctrans(GLIBC_2.2 .5) [2]
fputwc(GLIBC_2.2. 5) [2]	swscanf(GLIBC_2. 2.5) [2]	wscoll(GLIBC_2.2 .5) [2]	wctof(GLIBC_2.2. 5) [2]	wctype(GLIBC_2.2. 5) [2]
fputws(GLIBC_2.2. 5) [2]	towctrans(GLIBC_2 .2.5) [2]	wcscpy(GLIBC_2.2 .5) [2]	wctoimax(GLIBC_ 2.2.5) [2]	wcwidth(GLIBC_2. 2.5) [2]

fwidew(GLIBC_2.2.5) [2]	towlower(GLIBC_2.2.5) [2]	wscspn(GLIBC_2.2.5) [2]	wstok(GLIBC_2.2.5) [2]	wmemchr(GLIBC_2.2.5) [2]
fwprintf(GLIBC_2.2.5) [2]	toupper(GLIBC_2.2.5) [2]	wcsdup(GLIBC_2.2.5) [1]	wstol(GLIBC_2.2.5) [2]	wmemcmp(GLIBC_2.2.5) [2]
fwscanf(GLIBC_2.2.5) [2]	ungetwc(GLIBC_2.2.5) [2]	wcsftime(GLIBC_2.2.5) [2]	wstold(GLIBC_2.2.5) [2]	wmemcpy(GLIBC_2.2.5) [2]
getwc(GLIBC_2.2.5) [2]	vwprintf(GLIBC_2.2.5) [2]	wcslen(GLIBC_2.2.5) [2]	wstoll(GLIBC_2.2.5) [2]	wmemmove(GLIBC_2.2.5) [2]
getwchar(GLIBC_2.2.5) [2]	vwscanf(GLIBC_2.2.5) [2]	wcsncasecmp(GLIBC_2.2.5) [1]	wstombs(GLIBC_2.2.5) [2]	wmemset(GLIBC_2.2.5) [2]
mblen(GLIBC_2.2.5) [2]	vswprintf(GLIBC_2.2.5) [2]	wcsncat(GLIBC_2.2.5) [2]	wstoq(GLIBC_2.2.5) [1]	wprintf(GLIBC_2.2.5) [2]
mbrlen(GLIBC_2.2.5) [2]	vswscanf(GLIBC_2.2.5) [2]	wcsncmp(GLIBC_2.2.5) [2]	wstoul(GLIBC_2.2.5) [2]	wscanf(GLIBC_2.2.5) [2]
mbrtowc(GLIBC_2.2.5) [2]	vwprintf(GLIBC_2.2.5) [2]	wcsncpy(GLIBC_2.2.5) [2]	wstoull(GLIBC_2.2.5) [2]	

94

95 *Referenced Specification(s)*

96 [1]. this specification

97 [2]. ISO POSIX (2003)

1.2.8. String Functions

1.2.8.1. Interfaces for String Functions

99 An LSB conforming implementation shall provide the architecture specific functions for String Functions specified in
100 Table 1-13, with the full functionality as described in the referenced underlying specification.

101 **Table 1-13. libc - String Functions Function Interfaces**

__memcpy(GLIBC_2.2.5) [1]	bzero(GLIBC_2.2.5) [2]	strcasestr(GLIBC_2.2.5) [1]	strncasecmp(GLIBC_2.2.5) [2]	strtoimax(GLIBC_2.2.5) [2]
__rawmemchr(GLIBC_2.2.5) [1]	ffs(GLIBC_2.2.5) [2]	strcat(GLIBC_2.2.5) [2]	strncat(GLIBC_2.2.5) [2]	strtok(GLIBC_2.2.5) [2]
__stpcpy(GLIBC_2.2.5) [1]	index(GLIBC_2.2.5) [2]	strchr(GLIBC_2.2.5) [2]	strncmp(GLIBC_2.2.5) [2]	strtok_r(GLIBC_2.2.5) [2]
__strdup(GLIBC_2.2.5) [1]	memcpy(GLIBC_2.2.5) [2]	strcmp(GLIBC_2.2.5) [2]	strncpy(GLIBC_2.2.5) [2]	strtol(GLIBC_2.2.5) [2]
__strtod_internal(GLIBC_2.2.5) [1]	memchr(GLIBC_2.2.5) [2]	strcoll(GLIBC_2.2.5) [2]	strndup(GLIBC_2.2.5) [1]	strtoll(GLIBC_2.2.5) [2]
__strtof_internal(GLIBC_2.2.5) [1]	memcmp(GLIBC_2.2.5) [2]	strcpy(GLIBC_2.2.5) [2]	strnlen(GLIBC_2.2.5) [1]	strtoq(GLIBC_2.2.5) [2]

LIBC_2.2.5) [1]	.2.5) [2]	5) [2]	5) [1]) [1]
__strtok_r(GLIBC_2.2.5) [1]	memcpy(GLIBC_2.2.5) [2]	strcspn(GLIBC_2.2.5) [2]	strpbrk(GLIBC_2.2.5) [2]	strtoull(GLIBC_2.2.5) [2]
__strtol_internal(GLIBC_2.2.5) [1]	memmove(GLIBC_2.2.5) [2]	strdup(GLIBC_2.2.5) [2]	strptime(GLIBC_2.2.5) [1]	strtoumax(GLIBC_2.2.5) [2]
__strtold_internal(GLIBC_2.2.5) [1]	memchr(GLIBC_2.2.5) [1]	strerror(GLIBC_2.2.5) [2]	strchr(GLIBC_2.2.5) [2]	strtouq(GLIBC_2.2.5) [1]
__strtol_internal(GLIBC_2.2.5) [1]	memset(GLIBC_2.2.5) [2]	strerror_r(GLIBC_2.2.5) [1]	strsep(GLIBC_2.2.5) [1]	strverscmp(GLIBC_2.2.5) [1]
__strtoul_internal(GLIBC_2.2.5) [1]	rindex(GLIBC_2.2.5) [2]	strfmon(GLIBC_2.2.5) [2]	strsignal(GLIBC_2.2.5) [1]	strxfrm(GLIBC_2.2.5) [2]
__strtoull_internal(GLIBC_2.2.5) [1]	stpcpy(GLIBC_2.2.5) [1]	strfry(GLIBC_2.2.5) [1]	strspn(GLIBC_2.2.5) [2]	swab(GLIBC_2.2.5) [2]
bcmp(GLIBC_2.2.5) [2]	stpncpy(GLIBC_2.2.5) [1]	strftime(GLIBC_2.2.5) [2]	strstr(GLIBC_2.2.5) [2]	
bcopy(GLIBC_2.2.5) [2]	strcasestr(GLIBC_2.2.5) [2]	strlen(GLIBC_2.2.5) [2]	strtof(GLIBC_2.2.5) [2]	

102

103 *Referenced Specification(s)*

104 [1]. this specification

105 [2]. ISO POSIX (2003)

1.2.9. IPC Functions

1.2.9.1. Interfaces for IPC Functions

107 An LSB conforming implementation shall provide the architecture specific functions for IPC Functions specified in
108 Table 1-14, with the full functionality as described in the referenced underlying specification.

109 **Table 1-14. libc - IPC Functions Function Interfaces**

ftok(GLIBC_2.2.5) [1]	msgrcv(GLIBC_2.2.5) [1]	semget(GLIBC_2.2.5) [1]	shmctl(GLIBC_2.2.5) [1]	
msgctl(GLIBC_2.2.5) [1]	msgsnd(GLIBC_2.2.5) [1]	semop(GLIBC_2.2.5) [1]	shmdt(GLIBC_2.2.5) [1]	
msgget(GLIBC_2.2.5) [1]	semctl(GLIBC_2.2.5) [1]	shmat(GLIBC_2.2.5) [1]	shmget(GLIBC_2.2.5) [1]	

110

111 *Referenced Specification(s)*

112 [1]. ISO POSIX (2003)

1.2.10. Regular Expressions

1.2.10.1. Interfaces for Regular Expressions

114 An LSB conforming implementation shall provide the architecture specific functions for Regular Expressions
115 specified in Table 1-15, with the full functionality as described in the referenced underlying specification.

116 **Table 1-15. libc - Regular Expressions Function Interfaces**

117 regcomp(GLIBC_2.2.5) [1]	regerror(GLIBC_2.2.5) [1]	regexec(GLIBC_2.2.5) [1]	regfree(GLIBC_2.2.5) [1]	
------------------------------	---------------------------	--------------------------	--------------------------	--

118 *Referenced Specification(s)*

119 [1]. ISO POSIX (2003)

120 An LSB conforming implementation shall provide the architecture specific deprecated functions for Regular
121 Expressions specified in Table 1-16, with the full functionality as described in the referenced underlying specification.

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

124 **Table 1-16. libc - Regular Expressions Deprecated Function Interfaces**

125 advance(GLIBC_2.2.5) [1]	re_comp(GLIBC_2.2.5) [1]	re_exec(GLIBC_2.2.5) [1]	step(GLIBC_2.2.5) [1]	
------------------------------	--------------------------	--------------------------	-----------------------	--

126 *Referenced Specification(s)*

127 [1]. SUSv2

128 An LSB conforming implementation shall provide the architecture specific deprecated data interfaces for Regular
129 Expressions specified in Table 1-17, with the full functionality as described in the referenced underlying specification.

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

132 **Table 1-17. libc - Regular Expressions Deprecated Data Interfaces**

133 loc1(GLIBC_2.2.5) [1]	loc2(GLIBC_2.2.5) [1]	locs(GLIBC_2.2.5) [1]		
---------------------------	-----------------------	-----------------------	--	--

134 *Referenced Specification(s)*

135 [1]. SUSv2

1.2.11. Character Type Functions

1.2.11.1. Interfaces for Character Type Functions

137 An LSB conforming implementation shall provide the architecture specific functions for Character Type Functions
138 specified in Table 1-18, with the full functionality as described in the referenced underlying specification.

139 **Table 1-18. libc - Character Type Functions Function Interfaces**

__ctype_get_mb_cur_max(GLIBC_2.2.5) [1]	isdigit(GLIBC_2.2.5) [2]	iswalnum(GLIBC_2.2.5) [2]	iswlower(GLIBC_2.2.5) [2]	toascii(GLIBC_2.2.5) [2]
_tolower(GLIBC_2.2.5) [2]	isgraph(GLIBC_2.2.5) [2]	iswalphabet(GLIBC_2.2.5) [2]	iswprint(GLIBC_2.2.5) [2]	tolower(GLIBC_2.2.5) [2]
_toupper(GLIBC_2.2.5) [2]	islower(GLIBC_2.2.5) [2]	iswblank(GLIBC_2.2.5) [2]	iswpunct(GLIBC_2.2.5) [2]	toupper(GLIBC_2.2.5) [2]
isalnum(GLIBC_2.2.5) [2]	isprint(GLIBC_2.2.5) [2]	iswcntrl(GLIBC_2.2.5) [2]	iswspace(GLIBC_2.2.5) [2]	
isalpha(GLIBC_2.2.5) [2]	ispunct(GLIBC_2.2.5) [2]	iswctype(GLIBC_2.2.5) [2]	iswupper(GLIBC_2.2.5) [2]	
isascii(GLIBC_2.2.5) [2]	isspace(GLIBC_2.2.5) [2]	iswdigit(GLIBC_2.2.5) [2]	iswxdigit(GLIBC_2.2.5) [2]	
isctrl(GLIBC_2.2.5) [2]	isupper(GLIBC_2.2.5) [2]	iswgraph(GLIBC_2.2.5) [2]	isxdigit(GLIBC_2.2.5) [2]	

140

141 *Referenced Specification(s)*

142 [1]. this specification

143 [2]. ISO POSIX (2003)

1.2.12. Time Manipulation

1.2.12.1. Interfaces for Time Manipulation

145 An LSB conforming implementation shall provide the architecture specific functions for Time Manipulation specified
146 in Table 1-19, with the full functionality as described in the referenced underlying specification.

147 **Table 1-19. libc - Time Manipulation Function Interfaces**

adjtime(GLIBC_2.2.5) [1]	ctime(GLIBC_2.2.5) [2]	gmtime(GLIBC_2.2.5) [2]	localtime_r(GLIBC_2.2.5) [2]	ualarm(GLIBC_2.2.5) [2]
asctime(GLIBC_2.2.5) [2]	ctime_r(GLIBC_2.2.5) [2]	gmtime_r(GLIBC_2.2.5) [2]	mktime(GLIBC_2.2.5) [2]	
asctime_r(GLIBC_2.2.5) [2]	difftime(GLIBC_2.2.5) [2]	localtime(GLIBC_2.2.5) [2]	tzset(GLIBC_2.2.5) [2]	

148

149 *Referenced Specification(s)*

150 [1]. this specification

151 [2]. ISO POSIX (2003)

152 An LSB conforming implementation shall provide the architecture specific deprecated functions for Time
 153 Manipulation specified in Table 1-20, with the full functionality as described in the referenced underlying
 154 specification.

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

157 **Table 1-20. libc - Time Manipulation Deprecated Function Interfaces**

adjtimex(GLIBC_2.2.5) [1]				
---------------------------	--	--	--	--

158
 159 *Referenced Specification(s)*

160 [1]. this specification

161 An LSB conforming implementation shall provide the architecture specific data interfaces for Time Manipulation
 162 specified in Table 1-21, with the full functionality as described in the referenced underlying specification.

163 **Table 1-21. libc - Time Manipulation Data Interfaces**

__daylight(GLIBC_2.2.5) [1]	__tzname(GLIBC_2.2.5) [1]	timezone(GLIBC_2.2.5) [2]		
__timezone(GLIBC_2.2.5) [1]	daylight(GLIBC_2.2.5) [2]	tzname(GLIBC_2.2.5) [2]		

164
 165 *Referenced Specification(s)*

166 [1]. this specification

167 [2]. ISO POSIX (2003)

1.2.13. Terminal Interface Functions

1.2.13.1. Interfaces for Terminal Interface Functions

168 An LSB conforming implementation shall provide the architecture specific functions for Terminal Interface Functions
 169 specified in Table 1-22, with the full functionality as described in the referenced underlying specification.

171 **Table 1-22. libc - Terminal Interface Functions Function Interfaces**

cfgetispeed(GLIBC_2.2.5) [1]	cfsetispeed(GLIBC_2.2.5) [1]	tcdrain(GLIBC_2.2.5) [1]	tcgetattr(GLIBC_2.2.5) [1]	tcsendbreak(GLIBC_2.2.5) [1]
cfgetospeed(GLIBC_2.2.5) [1]	cfsetospeed(GLIBC_2.2.5) [1]	tcflow(GLIBC_2.2.5) [1]	tcgetpgrp(GLIBC_2.2.5) [1]	tcsetattr(GLIBC_2.2.5) [1]
cfmakeraw(GLIBC_2.2.5) [2]	cfsetspeed(GLIBC_2.2.5) [2]	tcflush(GLIBC_2.2.5) [1]	tcgetsid(GLIBC_2.2.5) [1]	tcsetpgrp(GLIBC_2.2.5) [1]

172
 173 *Referenced Specification(s)*

174 [1]. ISO POSIX (2003)

175 [2]. this specification

1.2.14. System Database Interface

1.2.14.1. Interfaces for System Database Interface

177 An LSB conforming implementation shall provide the architecture specific functions for System Database Interface
178 specified in Table 1-23, with the full functionality as described in the referenced underlying specification.

179 **Table 1-23. libc - System Database Interface Function Interfaces**

endgrent(GLIBC_2.2.5) [1]	getgrgid(GLIBC_2.2.5) [1]	getprotobynumber(GLIBC_2.2.5) [1]	getservbyport(GLIBC_2.2.5) [1]	setgrent(GLIBC_2.2.5) [1]
endnetent(GLIBC_2.2.5) [1]	getgrgid_r(GLIBC_2.2.5) [1]	getprotoent(GLIBC_2.2.5) [1]	getservent(GLIBC_2.2.5) [1]	setgroups(GLIBC_2.2.5) [2]
endprotoent(GLIBC_2.2.5) [1]	getgrnam(GLIBC_2.2.5) [1]	getpwent(GLIBC_2.2.5) [1]	getutent(GLIBC_2.2.5) [2]	setnetent(GLIBC_2.2.5) [1]
endpwent(GLIBC_2.2.5) [1]	getgrnam_r(GLIBC_2.2.5) [1]	getpwnam(GLIBC_2.2.5) [1]	getutent_r(GLIBC_2.2.5) [2]	setprotoent(GLIBC_2.2.5) [1]
endservent(GLIBC_2.2.5) [1]	gethostbyaddr(GLIBC_2.2.5) [1]	getpwnam_r(GLIBC_2.2.5) [1]	getutxent(GLIBC_2.2.5) [1]	setpwent(GLIBC_2.2.5) [1]
endutent(GLIBC_2.2.5) [3]	gethostbyname(GLIBC_2.2.5) [1]	getpwuid(GLIBC_2.2.5) [1]	getutxid(GLIBC_2.2.5) [1]	setservent(GLIBC_2.2.5) [1]
endutxent(GLIBC_2.2.5) [1]	getnetbyaddr(GLIBC_2.2.5) [1]	getpwuid_r(GLIBC_2.2.5) [1]	getutxline(GLIBC_2.2.5) [1]	setutent(GLIBC_2.2.5) [2]
getgrent(GLIBC_2.2.5) [1]	getprotobyname(GLIBC_2.2.5) [1]	getservbyname(GLIBC_2.2.5) [1]	pututxline(GLIBC_2.2.5) [1]	setutxent(GLIBC_2.2.5) [1]

180

181 *Referenced Specification(s)*

182 [1]. ISO POSIX (2003)

183 [2]. this specification

184 [3]. SUSv2

1.2.15. Language Support

1.2.15.1. Interfaces for Language Support

186 An LSB conforming implementation shall provide the architecture specific functions for Language Support specified
187 in Table 1-24, with the full functionality as described in the referenced underlying specification.

188 **Table 1-24. libc - Language Support Function Interfaces**

__libc_start_main(GLIBC_2.2.5) [1]	_obstack_begin(GLIBC_2.2.5) [1]	_obstack_newchunk(GLIBC_2.2.5) [1]	obstack_free(GLIBC_2.2.5) [1]	
------------------------------------	---------------------------------	------------------------------------	-------------------------------	--

189

190 *Referenced Specification(s)*

191 [1]. this specification

1.2.16. Large File Support

1.2.16.1. Interfaces for Large File Support

193 An LSB conforming implementation shall provide the architecture specific functions for Large File Support specified
194 in Table 1-25, with the full functionality as described in the referenced underlying specification.

195 **Table 1-25. libc - Large File Support Function Interfaces**

__fxstat64(GLIBC_2.2.5) [1]	fopen64(GLIBC_2.2.5) [2]	ftello64(GLIBC_2.2.5) [2]	lseek64(GLIBC_2.2.5) [2]	readdir64(GLIBC_2.2.5) [2]
__lxstat64(GLIBC_2.2.5) [1]	freopen64(GLIBC_2.2.5) [2]	ftruncate64(GLIBC_2.2.5) [2]	mkstemp64(GLIBC_2.2.5) [2]	statvfs64(GLIBC_2.2.5) [2]
__xstat64(GLIBC_2.2.5) [1]	fseeko64(GLIBC_2.2.5) [2]	ftw64(GLIBC_2.2.5) [2]	mmap64(GLIBC_2.2.5) [2]	tmpfile64(GLIBC_2.2.5) [2]
creat64(GLIBC_2.2.5) [2]	fsetpos64(GLIBC_2.2.5) [2]	getrlimit64(GLIBC_2.2.5) [2]	nftw64(GLIBC_2.3) [2]	truncate64(GLIBC_2.2.5) [2]
fgetpos64(GLIBC_2.2.5) [2]	fstatvfs64(GLIBC_2.2.5) [2]	lockf64(GLIBC_2.2.5) [2]	open64(GLIBC_2.2.5) [2]	

196

197 *Referenced Specification(s)*

198 [1]. this specification

199 [2]. Large File Support

1.2.17. Standard Library

1.2.17.1. Interfaces for Standard Library

201 An LSB conforming implementation shall provide the architecture specific functions for Standard Library specified in
202 Table 1-26, with the full functionality as described in the referenced underlying specification.

203 **Table 1-26. libc - Standard Library Function Interfaces**

_Exit(GLIBC_2.2.5) [1]	dirname(GLIBC_2.2.5) [1]	glob(GLIBC_2.2.5) [1]	lsearch(GLIBC_2.2.5) [1]	srand(GLIBC_2.2.5) [1]
__assert_fail(GLIBC_2.2.5) [2]	div(GLIBC_2.2.5) [1]	glob64(GLIBC_2.2.5) [2]	makecontext(GLIBC_2.2.5) [1]	srand48(GLIBC_2.2.5) [1]
__cxa_atexit(GLIBC_2.2.5) [2]	drand48(GLIBC_2.2.5) [1]	globfree(GLIBC_2.2.5) [1]	malloc(GLIBC_2.2.5) [1]	srandom(GLIBC_2.2.5) [1]
__errno_location(GLIBC_2.2.5) [2]	ecvt(GLIBC_2.2.5) [1]	globfree64(GLIBC_2.2.5) [2]	memmem(GLIBC_2.2.5) [2]	strtod(GLIBC_2.2.5) [1]

__fpending(GLIBC_2.2.5) [2]	erand48(GLIBC_2.2.5) [1]	grantpt(GLIBC_2.2.5) [1]	mkstemp(GLIBC_2.2.5) [1]	strtol(GLIBC_2.2.5) [1]
__getpagesize(GLIBC_2.2.5) [2]	err(GLIBC_2.2.5) [2]	hcreate(GLIBC_2.2.5) [1]	mktemp(GLIBC_2.2.5) [1]	strtoul(GLIBC_2.2.5) [1]
__isinf(GLIBC_2.2.5) [2]	error(GLIBC_2.2.5) [2]	hdestroy(GLIBC_2.2.5) [1]	mrnd48(GLIBC_2.2.5) [1]	swapcontext(GLIBC_2.2.5) [1]
__isinf(GLIBC_2.2.5) [2]	errx(GLIBC_2.2.5) [2]	hsearch(GLIBC_2.2.5) [1]	nftw(GLIBC_2.3.3) [1]	syslog(GLIBC_2.2.5) [1]
__isinfl(GLIBC_2.2.5) [2]	fcvt(GLIBC_2.2.5) [1]	htonl(GLIBC_2.2.5) [1]	nrnd48(GLIBC_2.2.5) [1]	system(GLIBC_2.2.5) [2]
__isnan(GLIBC_2.2.5) [2]	fmtmsg(GLIBC_2.2.5) [1]	htons(GLIBC_2.2.5) [1]	ntohl(GLIBC_2.2.5) [1]	tdelete(GLIBC_2.2.5) [1]
__isnanf(GLIBC_2.2.5) [2]	fnmatch(GLIBC_2.2.5) [1]	imaxabs(GLIBC_2.2.5) [1]	ntohs(GLIBC_2.2.5) [1]	tfind(GLIBC_2.2.5) [1]
__isnans(GLIBC_2.2.5) [2]	fpathconf(GLIBC_2.2.5) [1]	imaxdiv(GLIBC_2.2.5) [1]	openlog(GLIBC_2.2.5) [1]	tmpfile(GLIBC_2.2.5) [1]
__sysconf(GLIBC_2.2.5) [2]	free(GLIBC_2.2.5) [1]	inet_addr(GLIBC_2.2.5) [1]	perror(GLIBC_2.2.5) [1]	tmpnam(GLIBC_2.2.5) [1]
_exit(GLIBC_2.2.5) [1]	freeaddrinfo(GLIBC_2.2.5) [1]	inet_ntoa(GLIBC_2.2.5) [1]	posix_memalign(GLIBC_2.2.5) [1]	tsearch(GLIBC_2.2.5) [1]
_longjmp(GLIBC_2.2.5) [1]	ftrylockfile(GLIBC_2.2.5) [1]	inet_ntop(GLIBC_2.2.5) [1]	ptsname(GLIBC_2.2.5) [1]	ttynam(GLIBC_2.2.5) [1]
_setjmp(GLIBC_2.2.5) [1]	ftw(GLIBC_2.2.5) [1]	inet_pton(GLIBC_2.2.5) [1]	putenv(GLIBC_2.2.5) [1]	ttynam_r(GLIBC_2.2.5) [1]
a64l(GLIBC_2.2.5) [1]	funlockfile(GLIBC_2.2.5) [1]	initstate(GLIBC_2.2.5) [1]	qsort(GLIBC_2.2.5) [1]	twalk(GLIBC_2.2.5) [1]
abort(GLIBC_2.2.5) [1]	gai_strerror(GLIBC_2.2.5) [1]	insque(GLIBC_2.2.5) [1]	rand(GLIBC_2.2.5) [1]	unlockpt(GLIBC_2.2.5) [1]
abs(GLIBC_2.2.5) [1]	gcvt(GLIBC_2.2.5) [1]	isatty(GLIBC_2.2.5) [1]	rand_r(GLIBC_2.2.5) [1]	unsetenv(GLIBC_2.2.5) [1]
atof(GLIBC_2.2.5) [1]	getaddrinfo(GLIBC_2.2.5) [1]	isblank(GLIBC_2.2.5) [1]	random(GLIBC_2.2.5) [1]	usleep(GLIBC_2.2.5) [1]
atoi(GLIBC_2.2.5) [1]	getcwd(GLIBC_2.2.5) [1]	jrand48(GLIBC_2.2.5) [1]	random_r(GLIBC_2.2.5) [2]	verrx(GLIBC_2.2.5) [2]
atol(GLIBC_2.2.5) [1]	getdate(GLIBC_2.2.5) [1]	l64a(GLIBC_2.2.5) [1]	realloc(GLIBC_2.2.5) [1]	vfscanf(GLIBC_2.2.5) [1]
atoll(GLIBC_2.2.5) [1]	getenv(GLIBC_2.2.5) [1]	labs(GLIBC_2.2.5) [1]	realpath(GLIBC_2.2.5) [1]	vscanf(GLIBC_2.2.5) [1]

[1]	5) [1]	[1]	3) [1]	5) [1]
basename(GLIBC_2.2.5) [1]	getlogin(GLIBC_2.2.5) [1]	lcong48(GLIBC_2.2.5) [1]	remque(GLIBC_2.2.5) [1]	vsscanf(GLIBC_2.2.5) [1]
bsearch(GLIBC_2.2.5) [1]	getnameinfo(GLIBC_2.2.5) [1]	ldiv(GLIBC_2.2.5) [1]	seed48(GLIBC_2.2.5) [1]	vsyslog(GLIBC_2.2.5) [2]
calloc(GLIBC_2.2.5) [1]	getopt(GLIBC_2.2.5) [2]	lfind(GLIBC_2.2.5) [1]	setenv(GLIBC_2.2.5) [1]	warn(GLIBC_2.2.5) [2]
closelog(GLIBC_2.2.5) [1]	getopt_long(GLIBC_2.2.5) [2]	llabs(GLIBC_2.2.5) [1]	sethostid(GLIBC_2.2.5) [2]	warnx(GLIBC_2.2.5) [2]
confstr(GLIBC_2.2.5) [1]	getopt_long_only(GLIBC_2.2.5) [2]	lldiv(GLIBC_2.2.5) [1]	sethostname(GLIBC_2.2.5) [2]	wordexp(GLIBC_2.2.5) [1]
cuserid(GLIBC_2.2.5) [3]	getsubopt(GLIBC_2.2.5) [1]	longjmp(GLIBC_2.2.5) [1]	setlogmask(GLIBC_2.2.5) [1]	wordfree(GLIBC_2.2.5) [1]
daemon(GLIBC_2.2.5) [2]	gettimeofday(GLIBC_2.2.5) [1]	lrand48(GLIBC_2.2.5) [1]	setstate(GLIBC_2.2.5) [1]	

204

205 *Referenced Specification(s)*

206 [1]. ISO POSIX (2003)

207 [2]. this specification

208 [3]. SUSv2

209 An LSB conforming implementation shall provide the architecture specific data interfaces for Standard Library
 210 specified in Table 1-27, with the full functionality as described in the referenced underlying specification.

211 **Table 1-27. libc - Standard Library Data Interfaces**

__environ(GLIBC_2.2.5) [1]	_sys_errlist(GLIBC_2.3) [1]	getdate_err(GLIBC_2.2.5) [2]	opterr(GLIBC_2.2.5) [1]	optopt(GLIBC_2.2.5) [1]
_environ(GLIBC_2.2.5) [1]	environ(GLIBC_2.2.5) [2]	optarg(GLIBC_2.2.5) [2]	optind(GLIBC_2.2.5) [1]	

212

213 *Referenced Specification(s)*

214 [1]. this specification

215 [2]. ISO POSIX (2003)

1.3. Data Definitions for libc

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

219 These definitions are intended to supplement those provided in the referenced underlying specifications.

220 This specification uses ISO/IEC 9899 C Language as the reference programming language, and data definitions are
 221 specified in ISO C format. The C language is used here as a convenient notation. Using a C language description of
 222 these data objects does not preclude their use by other programming languages.

1.3.1. errno.h

```
223
224 #define EDEADLOCK          EDEADLK
```

1.3.2. inttypes.h

```
225
226 typedef long intmax_t;
227 typedef unsigned long uintptr_t;
228 typedef unsigned long uintmax_t;
229 typedef unsigned long uint64_t;
```

1.3.3. limits.h

```
230
231 #define LONG_MAX           0x7FFFFFFFFFFFFFFFL
232 #define ULONG_MAX         0xFFFFFFFFFFFFFFFFUL
233
234 #define CHAR_MAX           127
235 #define CHAR_MIN           SCHAR_MIN
```

1.3.4. setjmp.h

```
236
237 typedef long __jmp_buf[8];
```

1.3.5. signal.h

```
238
239 struct sigaction
240 {
241     union
242     {
243         sighandler_t _sa_handler;
244         void (*_sa_sigaction) (int, siginfo_t *, void *);
245     }
246     __sigaction_handler;
247     sigset_t sa_mask;
248     int sa_flags;
249     void (*sa_restorer) (void);
250 }
251 ;
252 #define MINSIGSTKSZ       2048
253 #define SIGSTKSZ          8192
254
255 struct _fpxreg
```

```

256  {
257    unsigned short significand[4];
258    unsigned short exponent;
259    unsigned short padding[3];
260  }
261  ;
262  struct _xmmreg
263  {
264    uint32_t element[4];
265  }
266  ;
267
268  struct _fpstate
269  {
270    uint16_t cwd;
271    uint16_t swd;
272    uint16_t ftw;
273    uint16_t fop;
274    uint64_t rip;
275    uint64_t rdp;
276    uint32_t mxcsr;
277    uint32_t mxcr_mask;
278    struct _fp_xreg _st[8];
279    struct _xmmreg _xmm[16];
280    uint32_t padding[24];
281  }
282  ;
283
284  struct sigcontext
285  {
286    unsigned long r8;
287    unsigned long r9;
288    unsigned long r10;
289    unsigned long r11;
290    unsigned long r12;
291    unsigned long r13;
292    unsigned long r14;
293    unsigned long r15;
294    unsigned long rdi;
295    unsigned long rsi;
296    unsigned long rbp;
297    unsigned long rbx;
298    unsigned long rdx;
299    unsigned long rax;
300    unsigned long rcx;
301    unsigned long rsp;
302    unsigned long rip;
303    unsigned long eflags;
304    unsigned short cs;
305    unsigned short gs;
306    unsigned short fs;
307    unsigned short __pad0;
308    unsigned long err;

```

```

309     unsigned long trapno;
310     unsigned long oldmask;
311     unsigned long cr2;
312     struct _fpstate *fpstate;
313     unsigned long __reserved1[8];
314 }
315 ;

```

1.3.6. stddef.h

```

316
317 typedef long ptrdiff_t;
318 typedef unsigned long size_t;

```

1.3.7. sys/ioctl.h

```

319
320 #define FIONREAD          0x541B
321 #define TIOCNOTTY       21538

```

1.3.8. sys/ipc.h

```

322
323 struct ipc_perm
324 {
325     key_t __key;
326     uid_t uid;
327     gid_t gid;
328     uid_t cuid;
329     uid_t cgid;
330     unsigned short mode;
331     unsigned short __pad1;
332     unsigned short __seq;
333     unsigned short __pad2;
334     unsigned long __unused1;
335     unsigned long __unused2;
336 }
337 ;

```

1.3.9. sys/mman.h

```

338
339 #define MCL_CURRENT      1
340 #define MCL_FUTURE      2

```

1.3.10. sys/msg.h

```

341
342 typedef unsigned long msgqnum_t;
343 typedef unsigned long msglen_t;
344

```

```

345 struct msqid_ds
346 {
347     struct ipc_perm msg_perm;
348     time_t msg_stime;
349     time_t msg_rtime;
350     time_t msg_ctime;
351     unsigned long __msg_cbytes;
352     msgqnum_t msg_qnum;
353     msglen_t msg_qbytes;
354     pid_t msg_lspid;
355     pid_t msg_lrpid;
356     unsigned long __unused4;
357     unsigned long __unused5;
358 }
359 ;

```

1.3.11. sys/sem.h

```

360
361 struct semid_ds
362 {
363     struct ipc_perm sem_perm;
364     time_t sem_otime;
365     unsigned long __unused1;
366     time_t sem_ctime;
367     unsigned long __unused2;
368     unsigned long sem_nsems;
369     unsigned long __unused3;
370     unsigned long __unused4;
371 }
372 ;

```

1.3.12. sys/shm.h

```

373
374 #define SHMLBA (__getpagesize())
375
376 typedef unsigned long shmatt_t;
377
378 struct shmid_ds
379 {
380     struct ipc_perm shm_perm;
381     size_t shm_segsz;
382     time_t shm_atime;
383     time_t shm_dtime;
384     time_t shm_ctime;
385     pid_t shm_cpid;
386     pid_t shm_lpid;
387     shmatt_t shm_nattch;
388     unsigned long __unused4;
389     unsigned long __unused5;
390 }

```

```
391 ;
```

1.3.13. sys/socket.h

```
392
393 typedef uint64_t __ss_aligntype;
```

1.3.14. sys/stat.h

```
394
395 #define _STAT_VER          1
396
397 struct stat
398 {
399     dev_t st_dev;
400     ino_t st_ino;
401     nlink_t st_nlink;
402     mode_t st_mode;
403     uid_t st_uid;
404     gid_t st_gid;
405     int pad0;
406     dev_t st_rdev;
407     off_t st_size;
408     blksize_t st_blksize;
409     blkcnt_t st_blocks;
410     struct timespec st_atim;
411     struct timespec st_mtim;
412     struct timespec st_ctim;
413     unsigned long __unused[3];
414 }
415 ;
416 struct stat64
417 {
418     dev_t st_dev;
419     ino64_t st_ino;
420     nlink_t st_nlink;
421     mode_t st_mode;
422     uid_t st_uid;
423     gid_t st_gid;
424     int pad0;
425     dev_t st_rdev;
426     off_t st_size;
427     blksize_t st_blksize;
428     blkcnt64_t st_blocks;
429     struct timespec st_atim;
430     struct timespec st_mtim;
431     struct timespec st_ctim;
432     unsigned long __unused[3];
433 }
434 ;
```


1.3.15. sys/statvfs.h

```

435
436 struct statvfs64
437 {
438     unsigned long f_bsize;
439     unsigned long f_frsize;
440     fsblkcnt64_t f_blocks;
441     fsblkcnt64_t f_bfree;
442     fsblkcnt64_t f_bavail;
443     fsfilcnt64_t f_files;
444     fsfilcnt64_t f_ffree;
445     fsfilcnt64_t f_favail;
446     unsigned long f_fsid;
447     unsigned long f_flag;
448     unsigned long f_namemax;
449     int __f_spare[6];
450 }
451 ;
452 struct statvfs
453 {
454     unsigned long f_bsize;
455     unsigned long f_frsize;
456     fsblkcnt_t f_blocks;
457     fsblkcnt_t f_bfree;
458     fsblkcnt_t f_bavail;
459     fsfilcnt_t f_files;
460     fsfilcnt_t f_ffree;
461     fsfilcnt_t f_favail;
462     unsigned long f_fsid;
463     unsigned long f_flag;
464     unsigned long f_namemax;
465     int __f_spare[6];
466 }
467 ;

```

1.3.16. sys/types.h

```

468
469 typedef long int64_t;
470
471 typedef int64_t ssize_t;

```

1.3.17. termios.h

```

472
473 #define OLCUC    0000002
474 #define ONLCR   0000004
475 #define XCASE   0000004
476 #define NLDLY   0000400
477 #define CR1     0001000
478 #define IUCLC   0001000

```

```

479 #define CR2      0002000
480 #define CR3      0003000
481 #define CRDLY    0003000
482 #define TAB1     0004000
483 #define TAB2     0010000
484 #define TAB3     0014000
485 #define TABDLY   0014000
486 #define BS1      0020000
487 #define BSDLY    0020000
488 #define VT1      0040000
489 #define VTDLY    0040000
490 #define FF1      0100000
491 #define FFDLY    0100000
492
493 #define VSUSP     10
494 #define VEOL      11
495 #define VREPRINT  12
496 #define VDISCARD  13
497 #define VWERASE   14
498 #define VEOL2    16
499 #define VMIN      6
500 #define VSWTC     7
501 #define VSTART    8
502 #define VSTOP     9
503
504 #define IXON      0002000
505 #define IXOFF     0010000
506
507 #define CS6       0000020
508 #define CS7       0000040
509 #define CS8       0000060
510 #define CSIZE     0000060
511 #define CSTOPB    0000100
512 #define CREAD     0000200
513 #define PARENB    0000400
514 #define PARODD    0001000
515 #define HUPCL     0002000
516 #define CLOCAL    0004000
517 #define VTIME     5
518
519 #define ISIG      0000001
520 #define ICANON    0000002
521 #define ECHOE     0000020
522 #define ECHOK     0000040
523 #define ECHONL    0000100
524 #define NOFLSH    0000200
525 #define TOSTOP    0000400
526 #define ECHOCTL   0001000
527 #define ECHOPRT   0002000
528 #define ECHOKE    0004000
529 #define FLUSHO    0010000
530 #define PENDIN    0040000
531 #define IEXTEN    0100000

```

1.3.18. ucontext.h

```

532
533 struct _libc_fpxreg
534 {
535     unsigned short significand[4];
536     unsigned short exponent;
537     unsigned short padding[3];
538 }
539 ;
540
541 typedef long greg_t;
542 #define NGREG    23
543
544 typedef greg_t gregset_t[23];
545
546 struct _libc_xmmreg
547 {
548     uint32_t element[4];
549 }
550 ;
551 struct _libc_fpstate
552 {
553     uint16_t cwd;
554     uint16_t swd;
555     uint16_t ftw;
556     uint16_t fop;
557     uint64_t rip;
558     uint64_t rdp;
559     uint32_t mxcsr;
560     uint32_t mxcr_mask;
561     struct _libc_fpxreg _st[8];
562     struct _libc_xmmreg _xmm[16];
563     uint32_t padding[24];
564 }
565 ;
566 typedef struct _libc_fpstate *fpregset_t;
567
568 typedef struct
569 {
570     gregset_t gregs;
571     fpregset_t fpregs;
572     unsigned long __reserved1[8];
573 }
574 mcontext_t;
575
576 typedef struct ucontext
577 {
578     unsigned long uc_flags;
579     struct ucontext *uc_link;
580     stack_t uc_stack;
581     mcontext_t uc_mcontext;

```

```

582     sigset_t uc_sigmask;
583     struct _libc_fpstate __fpregs_mem;
584 }
585 ucontext_t;

```

1.3.19. unistd.h

```

586
587 typedef long intptr_t;

```

1.3.20. utmp.h

```

588
589 struct lastlog
590 {
591     int32_t ll_time;
592     char ll_line[UT_LINESIZE];
593     char ll_host[UT_HOSTSIZE];
594 }
595 ;
596
597 struct utmp
598 {
599     short ut_type;
600     pid_t ut_pid;
601     char ut_line[UT_LINESIZE];
602     char ut_id[4];
603     char ut_user[UT_NAMESIZE];
604     char ut_host[UT_HOSTSIZE];
605     struct exit_status ut_exit;
606     int ut_session;
607     struct
608     {
609         int32_t tv_sec;
610         int32_t tv_usec;
611     }
612     ut_tv;
613     int32_t ut_addr_v6[4];
614     char __unused[20];
615 }
616 ;

```

1.3.21. utmpx.h

```

617
618 struct utmpx
619 {
620     short ut_type;
621     pid_t ut_pid;
622     char ut_line[UT_LINESIZE];
623     char ut_id[4];
624     char ut_user[UT_NAMESIZE];

```

```

625     char ut_host[UT_HOSTSIZE];
626     struct exit_status ut_exit;
627     int32_t ut_session;
628     struct
629     {
630         int32_t tv_sec;
631         int32_t tv_usec;
632     }
633     ut_tv;
634     int32_t ut_addr_v6[4];
635     char __unused[20];
636 }
637 ;

```

1.4. Interfaces for libm

638 Table 1-28 defines the library name and shared object name for the libm library

639 **Table 1-28. libm Definition**

Library:	libm
SONAME:	libm.so.6

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

ISO C (1999)

SUSv2

642 ISO POSIX (2003)

1.4.1. Math

1.4.1.1. Interfaces for Math

644 An LSB conforming implementation shall provide the architecture specific functions for Math specified in Table 1-29,
645 with the full functionality as described in the referenced underlying specification.

646 **Table 1-29. libm - Math Function Interfaces**

acos(GLIBC_2.2.5) [1]	cexp(GLIBC_2.2.5) [1]	expf(GLIBC_2.2.5) [1]	jnf(GLIBC_2.2.5) [2]	remquof(GLIBC_2. 2.5) [1]
acosf(GLIBC_2.2.5) [1]	cexpf(GLIBC_2.2.5) [1]	expl(GLIBC_2.2.5) [1]	jnl(GLIBC_2.2.5) [2]	remquol(GLIBC_2. 2.5) [1]
acosh(GLIBC_2.2.5) [1]	cexpl(GLIBC_2.2.5) [1]	expm1(GLIBC_2.2. 5) [1]	ldexp(GLIBC_2.2.5) [1]	rint(GLIBC_2.2.5) [1]
acoshf(GLIBC_2.2. 5) [1]	cimag(GLIBC_2.2. 5) [1]	fabs(GLIBC_2.2.5) [1]	ldexpf(GLIBC_2.2. 5) [1]	rintf(GLIBC_2.2.5) [1]
acoshl(GLIBC_2.2.) [1]	cimagf(GLIBC_2.2.) [1]	fabsf(GLIBC_2.2.5) [1]	ldexpl(GLIBC_2.2.) [1]	rintl(GLIBC_2.2.5) [1]

5) [1]	5) [1]	[1]	5) [1]	[1]
acosl(GLIBC_2.2.5) [1]	cimagl(GLIBC_2.2.5) [1]	fabsl(GLIBC_2.2.5) [1]	lgamma(GLIBC_2.2.5) [1]	round(GLIBC_2.2.5) [1]
asin(GLIBC_2.2.5) [1]	clog(GLIBC_2.2.5) [1]	fdim(GLIBC_2.2.5) [1]	lgamma_r(GLIBC_2.2.5) [2]	roundf(GLIBC_2.2.5) [1]
asinf(GLIBC_2.2.5) [1]	clog10(GLIBC_2.2.5) [2]	fdimf(GLIBC_2.2.5) [1]	lgammaf(GLIBC_2.2.5) [1]	roundl(GLIBC_2.2.5) [1]
asinh(GLIBC_2.2.5) [1]	clog10f(GLIBC_2.2.5) [2]	fdiml(GLIBC_2.2.5) [1]	lgammaf_r(GLIBC_2.2.5) [2]	scalb(GLIBC_2.2.5) [1]
asinhf(GLIBC_2.2.5) [1]	clog10l(GLIBC_2.2.5) [2]	feclearexcept(GLIBC_2.2.5) [1]	lgammal(GLIBC_2.2.5) [1]	scalbf(GLIBC_2.2.5) [2]
asinhf(GLIBC_2.2.5) [1]	clogf(GLIBC_2.2.5) [1]	fegetenv(GLIBC_2.2.5) [1]	lgammal_r(GLIBC_2.2.5) [2]	scalbl(GLIBC_2.2.5) [2]
asinl(GLIBC_2.2.5) [1]	clogl(GLIBC_2.2.5) [1]	fegetexceptflag(GLIBC_2.2.5) [1]	llrint(GLIBC_2.2.5) [1]	scalbln(GLIBC_2.2.5) [1]
atan(GLIBC_2.2.5) [1]	conj(GLIBC_2.2.5) [1]	fegetround(GLIBC_2.2.5) [1]	llrintf(GLIBC_2.2.5) [1]	scalblnf(GLIBC_2.2.5) [1]
atan2(GLIBC_2.2.5) [1]	conjf(GLIBC_2.2.5) [1]	feholdexcept(GLIBC_2.2.5) [1]	llrintl(GLIBC_2.2.5) [1]	scalblnl(GLIBC_2.2.5) [1]
atan2f(GLIBC_2.2.5) [1]	conjl(GLIBC_2.2.5) [1]	feraiseexcept(GLIBC_2.2.5) [1]	llround(GLIBC_2.2.5) [1]	scalbn(GLIBC_2.2.5) [1]
atan2l(GLIBC_2.2.5) [1]	copysign(GLIBC_2.2.5) [1]	fesetenv(GLIBC_2.2.5) [1]	llroundf(GLIBC_2.2.5) [1]	scalbnf(GLIBC_2.2.5) [1]
atanf(GLIBC_2.2.5) [1]	copysignf(GLIBC_2.2.5) [1]	fesetexceptflag(GLIBC_2.2.5) [1]	llroundl(GLIBC_2.2.5) [1]	scalbnl(GLIBC_2.2.5) [1]
atanh(GLIBC_2.2.5) [1]	copysignl(GLIBC_2.2.5) [1]	fesetround(GLIBC_2.2.5) [1]	log(GLIBC_2.2.5) [1]	significantd(GLIBC_2.2.5) [2]
atanhf(GLIBC_2.2.5) [1]	cos(GLIBC_2.2.5) [1]	fetestexcept(GLIBC_2.2.5) [1]	log10(GLIBC_2.2.5) [1]	significantdf(GLIBC_2.2.5) [2]
atanhl(GLIBC_2.2.5) [1]	cosf(GLIBC_2.2.5) [1]	feupdateenv(GLIBC_2.2.5) [1]	log10f(GLIBC_2.2.5) [1]	significantdl(GLIBC_2.2.5) [2]
atanl(GLIBC_2.2.5) [1]	cosh(GLIBC_2.2.5) [1]	finite(GLIBC_2.2.5) [3]	log10l(GLIBC_2.2.5) [1]	sin(GLIBC_2.2.5) [1]
cabs(GLIBC_2.2.5) [1]	coshf(GLIBC_2.2.5) [1]	finitef(GLIBC_2.2.5) [2]	log1p(GLIBC_2.2.5) [1]	sincos(GLIBC_2.2.5) [2]
cabsf(GLIBC_2.2.5) [1]	coshl(GLIBC_2.2.5) [1]	finitel(GLIBC_2.2.5) [2]	logb(GLIBC_2.2.5) [1]	sincosf(GLIBC_2.2.5) [2]

cabsl(GLIBC_2.2.5) [1]	cosl(GLIBC_2.2.5) [1]	floor(GLIBC_2.2.5) [1]	logf(GLIBC_2.2.5) [1]	sincosl(GLIBC_2.2.5) [2]
cacos(GLIBC_2.2.5) [1]	cpow(GLIBC_2.2.5) [1]	floorf(GLIBC_2.2.5) [1]	logl(GLIBC_2.2.5) [1]	sinf(GLIBC_2.2.5) [1]
cacosf(GLIBC_2.2.5) [1]	cpowf(GLIBC_2.2.5) [1]	floorl(GLIBC_2.2.5) [1]	lrint(GLIBC_2.2.5) [1]	sinh(GLIBC_2.2.5) [1]
cacosh(GLIBC_2.2.5) [1]	cpowl(GLIBC_2.2.5) [1]	fma(GLIBC_2.2.5) [1]	lrintf(GLIBC_2.2.5) [1]	sinhf(GLIBC_2.2.5) [1]
cacoshf(GLIBC_2.2.5) [1]	cproj(GLIBC_2.2.5) [1]	fmaf(GLIBC_2.2.5) [1]	lrintl(GLIBC_2.2.5) [1]	sinhl(GLIBC_2.2.5) [1]
cacoshl(GLIBC_2.2.5) [1]	cprojf(GLIBC_2.2.5) [1]	fmal(GLIBC_2.2.5) [1]	lround(GLIBC_2.2.5) [1]	sinl(GLIBC_2.2.5) [1]
caosl(GLIBC_2.2.5) [1]	cprojl(GLIBC_2.2.5) [1]	fmax(GLIBC_2.2.5) [1]	lroundf(GLIBC_2.2.5) [1]	sqrt(GLIBC_2.2.5) [1]
carg(GLIBC_2.2.5) [1]	creal(GLIBC_2.2.5) [1]	fmaxf(GLIBC_2.2.5) [1]	lroundl(GLIBC_2.2.5) [1]	sqrtf(GLIBC_2.2.5) [1]
cargf(GLIBC_2.2.5) [1]	crealf(GLIBC_2.2.5) [1]	fmaxl(GLIBC_2.2.5) [1]	matherr(GLIBC_2.2.5) [2]	sqrtl(GLIBC_2.2.5) [1]
cargl(GLIBC_2.2.5) [1]	creall(GLIBC_2.2.5) [1]	fmin(GLIBC_2.2.5) [1]	modf(GLIBC_2.2.5) [1]	tan(GLIBC_2.2.5) [1]
casin(GLIBC_2.2.5) [1]	csin(GLIBC_2.2.5) [1]	fminf(GLIBC_2.2.5) [1]	modff(GLIBC_2.2.5) [1]	tanf(GLIBC_2.2.5) [1]
casinf(GLIBC_2.2.5) [1]	csinf(GLIBC_2.2.5) [1]	fminl(GLIBC_2.2.5) [1]	modfl(GLIBC_2.2.5) [1]	tanh(GLIBC_2.2.5) [1]
casinh(GLIBC_2.2.5) [1]	csinh(GLIBC_2.2.5) [1]	fmod(GLIBC_2.2.5) [1]	nan(GLIBC_2.2.5) [1]	tanhf(GLIBC_2.2.5) [1]
casinhf(GLIBC_2.2.5) [1]	csinhf(GLIBC_2.2.5) [1]	fmodf(GLIBC_2.2.5) [1]	nanf(GLIBC_2.2.5) [1]	tanhl(GLIBC_2.2.5) [1]
casinhl(GLIBC_2.2.5) [1]	csinhl(GLIBC_2.2.5) [1]	fmodl(GLIBC_2.2.5) [1]	nanl(GLIBC_2.2.5) [1]	tanl(GLIBC_2.2.5) [1]
casinl(GLIBC_2.2.5) [1]	csinl(GLIBC_2.2.5) [1]	frexp(GLIBC_2.2.5) [1]	nearbyint(GLIBC_2.2.5) [1]	tgamma(GLIBC_2.2.5) [1]
catan(GLIBC_2.2.5) [1]	csqrt(GLIBC_2.2.5) [1]	frexpf(GLIBC_2.2.5) [1]	nearbyintf(GLIBC_2.2.5) [1]	tgammaf(GLIBC_2.2.5) [1]
catanf(GLIBC_2.2.5) [1]	csqrtf(GLIBC_2.2.5) [1]	frexpl(GLIBC_2.2.5) [1]	nearbyintl(GLIBC_2.2.5) [1]	tgammal(GLIBC_2.2.5) [1]
catanh(GLIBC_2.2.5) [1]	csqrtl(GLIBC_2.2.5) [1]	gamma(GLIBC_2.2.5) [1]	nextafter(GLIBC_2.2.5) [1]	trunc(GLIBC_2.2.5) [1]

5) [1]) [1]	.5) [3]	2.5) [1]	[1]
catanhf(GLIBC_2.2.5) [1]	ctan(GLIBC_2.2.5) [1]	gammaf(GLIBC_2.2.5) [2]	nextafterf(GLIBC_2.2.5) [1]	truncf(GLIBC_2.2.5) [1]
catanh(GLIBC_2.2.5) [1]	ctanf(GLIBC_2.2.5) [1]	gammal(GLIBC_2.2.5) [2]	nextafterl(GLIBC_2.2.5) [1]	truncl(GLIBC_2.2.5) [1]
catanl(GLIBC_2.2.5) [1]	ctanh(GLIBC_2.2.5) [1]	hypot(GLIBC_2.2.5) [1]	nexttoward(GLIBC_2.2.5) [1]	y0(GLIBC_2.2.5) [1]
cbrt(GLIBC_2.2.5) [1]	ctanhf(GLIBC_2.2.5) [1]	hypotf(GLIBC_2.2.5) [1]	nexttowardf(GLIBC_2.2.5) [1]	y0f(GLIBC_2.2.5) [2]
cbrtf(GLIBC_2.2.5) [1]	ctanh(GLIBC_2.2.5) [1]	hypotl(GLIBC_2.2.5) [1]	nexttowardl(GLIBC_2.2.5) [1]	y0l(GLIBC_2.2.5) [2]
cbrtl(GLIBC_2.2.5) [1]	ctanl(GLIBC_2.2.5) [1]	ilogb(GLIBC_2.2.5) [1]	pow(GLIBC_2.2.5) [1]	y1(GLIBC_2.2.5) [1]
ccos(GLIBC_2.2.5) [1]	dremf(GLIBC_2.2.5) [2]	ilogbf(GLIBC_2.2.5) [1]	pow10(GLIBC_2.2.5) [2]	y1f(GLIBC_2.2.5) [2]
ccosf(GLIBC_2.2.5) [1]	dreml(GLIBC_2.2.5) [2]	ilogbl(GLIBC_2.2.5) [1]	pow10f(GLIBC_2.2.5) [2]	y1l(GLIBC_2.2.5) [2]
ccosh(GLIBC_2.2.5) [1]	erf(GLIBC_2.2.5) [1]	j0(GLIBC_2.2.5) [1]	pow10l(GLIBC_2.2.5) [2]	yn(GLIBC_2.2.5) [1]
ccoshf(GLIBC_2.2.5) [1]	erfc(GLIBC_2.2.5) [1]	j0f(GLIBC_2.2.5) [2]	powf(GLIBC_2.2.5) [1]	ynf(GLIBC_2.2.5) [2]
ccoshl(GLIBC_2.2.5) [1]	erfcf(GLIBC_2.2.5) [1]	j0l(GLIBC_2.2.5) [2]	powl(GLIBC_2.2.5) [1]	ynl(GLIBC_2.2.5) [2]
ccosl(GLIBC_2.2.5) [1]	erfc1(GLIBC_2.2.5) [1]	j1(GLIBC_2.2.5) [1]	remainder(GLIBC_2.2.5) [1]	
ceil(GLIBC_2.2.5) [1]	erff(GLIBC_2.2.5) [1]	j1f(GLIBC_2.2.5) [2]	remainderf(GLIBC_2.2.5) [1]	
ceilf(GLIBC_2.2.5) [1]	erfl(GLIBC_2.2.5) [1]	j1l(GLIBC_2.2.5) [2]	remainderl(GLIBC_2.2.5) [1]	
ceil(GLIBC_2.2.5) [1]	exp(GLIBC_2.2.5) [1]	jn(GLIBC_2.2.5) [1]	remquo(GLIBC_2.2.5) [1]	

647

648 *Referenced Specification(s)*649 **[1].** ISO POSIX (2003)650 **[2].** ISO C (1999)651 **[3].** SUSv2

652 An LSB conforming implementation shall provide the architecture specific data interfaces for Math specified in Table
653 1-30, with the full functionality as described in the referenced underlying specification.

654 **Table 1-30. libm - Math Data Interfaces**

655	signgam(GLIBC_2.2.5) [1]				
-----	--------------------------	--	--	--	--

656 *Referenced Specification(s)*

657 [1]. ISO POSIX (2003)

1.5. Interfaces for libpthread

658 Table 1-31 defines the library name and shared object name for the libpthread library

659 **Table 1-31. libpthread Definition**

660	Library:	libpthread
	SONAME:	libpthread.so.0

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

Large File Support

this specification

662 ISO POSIX (2003)

1.5.1. Realtime Threads

663 **1.5.1.1. Interfaces for Realtime Threads**

664 No external functions are defined for libpthread - Realtime Threads

1.5.2. Advanced Realtime Threads

665 **1.5.2.1. Interfaces for Advanced Realtime Threads**

666 No external functions are defined for libpthread - Advanced Realtime Threads

1.5.3. Posix Threads

667 **1.5.3.1. Interfaces for Posix Threads**

668 An LSB conforming implementation shall provide the architecture specific functions for Posix Threads specified in

669 Table 1-32, with the full functionality as described in the referenced underlying specification.

670 **Table 1-32. libpthread - Posix Threads Function Interfaces**

670	__pthread_cleanup_p op(GLIBC_2.2.5) [1]	pthread_cancel(GLI BC_2.2.5) [2]	pthread_join(GLIB C_2.2.5) [2]	pthread_rwlock_des troy(GLIBC_2.2.5) [2]	pthread_setconcurr ency(GLIBC_2.2.5) [2]
	__pthread_cleanup_p	pthread_cond_broad	pthread_key_create(pthread_rwlock_init	pthread_setspecific(

ush(GLIBC_2.2.5) [1]	cast(GLIBC_2.3.2) [2]	GLIBC_2.2.5) [2]	(GLIBC_2.2.5) [2]	GLIBC_2.2.5) [2]
pread(GLIBC_2.2.5) [2]	pthread_cond_destroy(GLIBC_2.3.2) [2]	pthread_key_delete(GLIBC_2.2.5) [2]	pthread_rwlock_rdlock(GLIBC_2.2.5) [2]	pthread_sigmask(GLIBC_2.2.5) [2]
pread64(GLIBC_2.2.5) [3]	pthread_cond_init(GLIBC_2.3.2) [2]	pthread_kill(GLIBC_2.2.5) [2]	pthread_rwlock_timedrdlock(GLIBC_2.2.5) [2]	pthread_testcancel(GLIBC_2.2.5) [2]
pthread_attr_destroy(GLIBC_2.2.5) [2]	pthread_cond_signal(GLIBC_2.3.2) [2]	pthread_mutex_destroy(GLIBC_2.2.5) [2]	pthread_rwlock_timedwrlock(GLIBC_2.2.5) [2]	pwrite(GLIBC_2.2.5) [2]
pthread_attr_getdetachstate(GLIBC_2.2.5) [2]	pthread_cond_timedwait(GLIBC_2.3.2) [2]	pthread_mutex_init(GLIBC_2.2.5) [2]	pthread_rwlock_tryrdlock(GLIBC_2.2.5) [2]	pwrite64(GLIBC_2.2.5) [3]
pthread_attr_getguardsize(GLIBC_2.2.5) [2]	pthread_cond_wait(GLIBC_2.3.2) [2]	pthread_mutex_lock(GLIBC_2.2.5) [2]	pthread_rwlock_trywrlock(GLIBC_2.2.5) [2]	sem_close(GLIBC_2.2.5) [2]
pthread_attr_getschedparam(GLIBC_2.2.5) [2]	pthread_condattr_destroy(GLIBC_2.2.5) [2]	pthread_mutex_trylock(GLIBC_2.2.5) [2]	pthread_rwlock_unlink(GLIBC_2.2.5) [2]	sem_destroy(GLIBC_2.2.5) [2]
pthread_attr_getstackaddr(GLIBC_2.2.5) [2]	pthread_condattr_getpshared(GLIBC_2.2.5) [2]	pthread_mutex_unlink(GLIBC_2.2.5) [2]	pthread_rwlock_writelock(GLIBC_2.2.5) [2]	sem_getvalue(GLIBC_2.2.5) [2]
pthread_attr_getstacksize(GLIBC_2.2.5) [2]	pthread_condattr_init(GLIBC_2.2.5) [2]	pthread_mutexattr_destroy(GLIBC_2.2.5) [2]	pthread_rwlockattr_destroy(GLIBC_2.2.5) [2]	sem_init(GLIBC_2.2.5) [2]
pthread_attr_init(GLIBC_2.2.5) [2]	pthread_condattr_setpshared(GLIBC_2.2.5) [2]	pthread_mutexattr_getpshared(GLIBC_2.2.5) [2]	pthread_rwlockattr_getpshared(GLIBC_2.2.5) [2]	sem_open(GLIBC_2.2.5) [2]
pthread_attr_setdetachstate(GLIBC_2.2.5) [2]	pthread_create(GLIBC_2.2.5) [2]	pthread_mutexattr_gettype(GLIBC_2.2.5) [2]	pthread_rwlockattr_init(GLIBC_2.2.5) [2]	sem_post(GLIBC_2.2.5) [2]
pthread_attr_setguardsize(GLIBC_2.2.5) [2]	pthread_detach(GLIBC_2.2.5) [2]	pthread_mutexattr_init(GLIBC_2.2.5) [2]	pthread_rwlockattr_setpshared(GLIBC_2.2.5) [2]	sem_timedwait(GLIBC_2.2.5) [2]
pthread_attr_setschedparam(GLIBC_2.2.5) [2]	pthread_equal(GLIBC_2.2.5) [2]	pthread_mutexattr_setpshared(GLIBC_2.2.5) [2]	pthread_self(GLIBC_2.2.5) [2]	sem_trywait(GLIBC_2.2.5) [2]
pthread_attr_setstackaddr(GLIBC_2.2.5) [2]	pthread_exit(GLIBC_2.2.5) [2]	pthread_mutexattr_settype(GLIBC_2.2.5) [2]	pthread_setcancelstate(GLIBC_2.2.5) [2]	sem_unlink(GLIBC_2.2.5) [2]

) [2]		5) [2]	[2]	
pthread_attr_setstacksize(GLIBC_2.2.5) [2]	pthread_getspecific(GLIBC_2.2.5) [2]	pthread_once(GLIBC_2.2.5) [2]	pthread_setcanceltype(GLIBC_2.2.5) [2]	sem_wait(GLIBC_2.2.5) [2]

671

672 *Referenced Specification(s)*

673 [1]. this specification

674 [2]. ISO POSIX (2003)

675 [3]. Large File Support

1.6. Interfaces for libgcc_s

676 Table 1-33 defines the library name and shared object name for the libgcc_s library

677 **Table 1-33. libgcc_s Definition**

Library:	libgcc_s
SONAME:	libgcc_s.so.1

678

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

680 this specification

1.6.1. Unwind Library

1.6.1.1. Interfaces for Unwind Library

682 An LSB conforming implementation shall provide the architecture specific functions for Unwind Library specified in

683 Table 1-34, with the full functionality as described in the referenced underlying specification.

684 **Table 1-34. libgcc_s - Unwind Library Function Interfaces**

_Unwind_DeleteException(GCC_3.0) [1]	_Unwind_GetDataRelBase(GCC_3.0) [1]	_Unwind_GetLanguageSpecificData(GCC_3.0) [1]	_Unwind_RaiseException(GCC_3.0) [1]	_Unwind_SetIP(GCC_3.0) [1]
_Unwind_Find_FDE(GCC_3.0) [1]	_Unwind_GetGR(GCC_3.0) [1]	_Unwind_GetRegionStart(GCC_3.0) [1]	_Unwind_Resume(GCC_3.0) [1]	
_Unwind_ForcedUnwind(GCC_3.0) [1]	_Unwind_GetIP(GCC_3.0) [1]	_Unwind_GetTextRelBase(GCC_3.0) [1]	_Unwind_SetGR(GCC_3.0) [1]	

685

686 *Referenced Specification(s)*

687 [1]. this specification

1.7. Interface Definitions for libgcc_s

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

690 Other interfaces listed above for libgcc_s shall behave as described in the referenced base document.

`_Unwind_DeleteException`

Name

691 `_Unwind_DeleteException` — private C++ error handling method

Synopsis

692 `void _Unwind_DeleteException((struct _Unwind_Exception *object));`

Description

693 `_Unwind_DeleteException` deletes the given exception *object*. If a given runtime resumes normal execution
694 after catching a foreign exception, it will not know how to delete that exception. Such an exception shall be deleted by
695 calling `_Unwind_DeleteException`. This is a convenience function that calls the function pointed to by the
696 *exception_cleanup* field of the exception header.

`_Unwind_Find_FDE`

Name

697 `_Unwind_Find_FDE` — private C++ error handling method

Synopsis

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

Description

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

`_Unwind_ForcedUnwind`

Name

700 `_Unwind_ForcedUnwind` — private C++ error handling method

Synopsis

```
701 _Unwind_Reason_Code _Unwind_ForcedUnwind((struct _Unwind_Exception *object),  
702 _Unwind_Stop_Fn stop, void *stop_parameter);
```

Description

703 `_Unwind_ForcedUnwind` raises an exception for forced unwinding, passing along the given exception *object*,
704 which should have its *exception_class* and *exception_cleanup* fields set. The exception *object* has been allocated by
705 the language-specific runtime, and has a language-specific format, except that it shall contain an `_Unwind_Exception`
706 struct.

707 Forced unwinding is a single-phase process. *stop* and *stop_parameter* control the termination of the unwind
708 process instead of the usual personality routine query. *stop* is called for each unwind frame, with the parameters
709 described for the usual personality routine below, plus an additional *stop_parameter*.

Return Value

710 When *stop* identifies the destination frame, it transfers control to the user code as appropriate without returning,
711 normally after calling `_Unwind_DeleteException`. If not, then it should return an `_Unwind_Reason_Code` value.

712 If *stop* returns any reason code other than `_URC_NO_REASON`, then the stack state is indeterminate from the point
713 of view of the caller of `_Unwind_ForcedUnwind`. Rather than attempt to return, therefore, the unwind library should
714 use the *exception_cleanup* entry in the exception, and then call `abort`.

715 `_URC_NO_REASON`

716 This is not the destination from. The unwind runtime will call frame's personality routine with the
717 `_UA_FORCE_UNWIND` and `_UA_CLEANUP_PHASE` flag set in *actions*, and then unwind to the next frame and call
718 the *stop* function again.

719 `_URC_END_OF_STACK`

720 In order to allow `_Unwind_ForcedUnwind` to perform special processing when it reaches the end of the stack,
721 the unwind runtime will call it after the last frame is rejected, with a `NULL` stack pointer in the context, and the
722 *stop* function shall catch this condition. It may return this code if it cannot handle end-of-stack.

723 `_URC_FATAL_PHASE2_ERROR`

724 The *stop* function may return this code for other fatal conditions like stack corruption.

`_Unwind_GetDataRelBase`

Name

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

Synopsis

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

Description

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

`_Unwind_GetGR`

Name

728 `_Unwind_GetGR` — private C++ error handling method

Synopsis

729 `_Unwind_Word _Unwind_GetGR((struct _Unwind_Context *context), int index);`

Description

730 `_Unwind_GetGR` returns data at *index* found in *context*. The register is identified by its index: 0 to 31 are for the
731 fixed registers, and 32 to 127 are for the stacked registers.

732 During the two phases of unwinding, only GR1 has a guaranteed value, which is the global pointer of the frame
733 referenced by the unwind *context*. If the register has its NAT bit set, the behavior is unspecified.

`_Unwind_GetIP`

Name

734 `_Unwind_GetIP` — private C++ error handling method

Synopsis

735 `_Unwind_Ptr _Unwind_GetIP((struct _Unwind_Context *context));`

Description

736 `_Unwind_GetIP` returns the instruction pointer value for the routine identified by the unwind *context*.

`_Unwind_GetLanguageSpecificData`

Name

737 `_Unwind_GetLanguageSpecificData` — private C++ error handling method

Synopsis

```
738 _Unwind_Ptr _Unwind_GetLanguageSpecificData((struct _Unwind_Context *context), uint  
739 value);
```

Description

740 `_Unwind_GetLanguageSpecificData` returns the address of the language specific data area for the current stack
741 frame.

`_Unwind_GetRegionStart`

Name

742 `_Unwind_GetRegionStart` — private C++ error handling method

Synopsis

```
743 _Unwind_Ptr _Unwind_GetRegionStart((struct _Unwind_Context *context));
```

Description

744 `_Unwind_GetRegionStart` routine returns the address (i.e., 0) of the beginning of the procedure or code fragment
745 described by the current unwind descriptor block.

`_Unwind_GetTextRelBase`

Name

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

Synopsis

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

Description

748 `_Unwind_GetTextRelBase` calls the abort method, then returns.

`_Unwind_RaiseException`

Name

749 `_Unwind_RaiseException` — private C++ error handling method

Synopsis

750 `_Unwind_Reason_Code _Unwind_RaiseException((struct _Unwind_Exception *object));`

Description

751 `_Unwind_RaiseException` raises an exception, passing along the given exception *object*, which should have its
 752 *exception_class* and *exception_cleanup* fields set. The exception object has been allocated by the
 753 language-specific runtime, and has a language-specific format, exception that it shall contain an
 754 `_Unwind_Exception`.

Return Value

755 `_Unwind_RaiseException` does not return unless an error condition is found. If an error condition occurs, an
 756 `_Unwind_Reason_Code` is returned:

757 `_URC_END_OF_STACK`

758 The unwinder encountered the end of the stack during phase one without finding a handler. The unwind runtime
 759 will not have modified the stack. The C++ runtime will normally call `uncaught_exception` in this case.

760 `_URC_FATAL_PHASE1_ERROR`

761 The unwinder encountered an unexpected error during phase one, because of something like stack corruption.
 762 The unwind runtime will not have modified the stack. The C++ runtime will normally call `terminate` in this
 763 case.

764 `_URC_FATAL_PHASE2_ERROR`

765 The unwinder encountered an unexpected error during phase two. This is usually a *throw*, which will call
 766 `terminate`.

`_Unwind_Resume`

Name

767 `_Unwind_Resume` — private C++ error handling method

Synopsis

```
768 void _Unwind_Resume((struct _Unwind_Exception *object));
```

Description

769 `_Unwind_Resume` resumes propagation of an existing exception *object*. A call to this routine is inserted as the end
770 of a landing pad that performs cleanup, but does not resume normal execution. It causes unwinding to proceed further.

`_Unwind_SetGR`

Name

771 `_Unwind_SetGR` — private C++ error handling method

Synopsis

```
772 void _Unwind_SetGR((struct _Unwind_Context *context), int index, uint value);
```

Description

773 `_Unwind_SetGR` sets the *value* of the register *indexed* for the routine identified by the unwind *context*.

`_Unwind_SetIP`

Name

774 `_Unwind_SetIP` — private C++ error handling method

Synopsis

```
775 void _Unwind_SetIP((struct _Unwind_Context *context), uint value);
```

Description

776 `_Unwind_SetIP` sets the *value* of the instruction pointer for the routine identified by the unwind *context*

1.8. Interfaces for libdl

777 Table 1-35 defines the library name and shared object name for the libdl library

778 **Table 1-35. libdl Definition**

Library:	libdl
SONAME:	libdl.so.2

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

this specification

781 ISO POSIX (2003)

1.8.1. Dynamic Loader

782 1.8.1.1. Interfaces for Dynamic Loader

783 An LSB conforming implementation shall provide the architecture specific functions for Dynamic Loader specified in
784 Table 1-36, with the full functionality as described in the referenced underlying specification.

785 **Table 1-36. libdl - Dynamic Loader Function Interfaces**

dladdr(GLIBC_2.2.5) [1]	dlclose(GLIBC_2.2.5) [2]	dlderror(GLIBC_2.2.5) [2]	dlopen(GLIBC_2.2.5) [1]	dlsym(GLIBC_2.2.5) [1]
-------------------------	--------------------------	---------------------------	-------------------------	------------------------

787 *Referenced Specification(s)*

788 [1]. this specification

789 [2]. ISO POSIX (2003)

1.9. Interfaces for libcrypt

790 Table 1-37 defines the library name and shared object name for the libcrypt library

791 **Table 1-37. libcrypt Definition**

Library:	libcrypt
SONAME:	libcrypt.so.1

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

794 ISO POSIX (2003)

1.9.1. Encryption

795 1.9.1.1. Interfaces for Encryption

796 An LSB conforming implementation shall provide the architecture specific functions for Encryption specified in Table
797 1-38, with the full functionality as described in the referenced underlying specification.

798 **Table 1-38. libcrypt - Encryption Function Interfaces**

crypt(GLIBC_2.2.5)	encrypt(GLIBC_2.2.5)	setkey(GLIBC_2.2.5)		
--------------------	----------------------	---------------------	--	--

799	[1]	.5) [1]	5) [1]		
-----	-----	---------	--------	--	--

800 *Referenced Specification(s)*

801 [1]. ISO POSIX (2003)

II. Utility Libraries

Chapter 2. Libraries

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

2.1. Interfaces for libz

4 Table 2-1 defines the library name and shared object name for the libz library

5 **Table 2-1. libz Definition**

Library:	libz
SONAME:	libz.so.1

2.1.1. Compression Library

2.1.1.1. Interfaces for Compression Library

8 No external functions are defined for libz - Compression Library

2.2. Interfaces for libncurses

9 Table 2-2 defines the library name and shared object name for the libncurses library

10 **Table 2-2. libncurses Definition**

Library:	libncurses
SONAME:	libncurses.so.5

2.2.1. Curses

2.2.1.1. Interfaces for Curses

13 No external functions are defined for libncurses - Curses

2.3. Interfaces for libutil

14 Table 2-3 defines the library name and shared object name for the libutil library

15 **Table 2-3. libutil Definition**

Library:	libutil
SONAME:	libutil.so.1

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

2.3.1. Utility Functions

2.3.1.1. Interfaces for Utility Functions

20 An LSB conforming implementation shall provide the architecture specific functions for Utility Functions specified in
 21 Table 2-4, with the full functionality as described in the referenced underlying specification.

22 **Table 2-4. libutil - Utility Functions Function Interfaces**

forkpty(GLIBC_2.2.5) [1]	login_tty(GLIBC_2.2.5) [1]	logwtmp(GLIBC_2.2.5) [1]		
login(GLIBC_2.2.5) [1]	logout(GLIBC_2.2.5) [1]	openpty(GLIBC_2.2.5) [1]		

24 *Referenced Specification(s)*

25 [1]. this specification

Appendix A. Alphabetical Listing of Interfaces

A.1. libgcc_s

- 1 The behaviour of the interfaces in this library is specified by the following Standards.
2 this specification

3 **Table A-1. libgcc_s Function Interfaces**

_Unwind_DeleteException[1]	_Unwind_GetIP[1]	_Unwind_Resume[1]
_Unwind_Find_FDE[1]	_Unwind_GetLanguageSpecificData[1]	_Unwind_SetGR[1]
_Unwind_ForcedUnwind[1]	_Unwind_GetRegionStart[1]	_Unwind_SetIP[1]
_Unwind_GetDataRelBase[1]	_Unwind_GetTextRelBase[1]	
_Unwind_GetGR[1]	_Unwind_RaiseException[1]	

4

Linux Packaging Specification

2

3 **Linux Packaging Specification**

Table of Contents

I. Package Format and Installation	49
1. Software Installation	1
1.1. Package Dependencies.....	1
1.2. Package Architecture Considerations	1

I. Package Format and Installation

Chapter 1. Software Installation

1.1. Package Dependencies

- 1 The LSB runtime environment shall provide the following dependencies.
- 2 `lsb-core-amd64`
 - 3 This dependency is used to indicate that the application is dependent on features contained in the LSB-Core
 - 4 specification.
- 5 Other LSB modules may add additional dependencies; such dependencies shall have the format `lsb-module-amd64`.

1.2. Package Architecture Considerations

- 6 All packages must specify an architecture of `x86_64`. An LSB runtime environment must accept an architecture of
- 7 `x86_64` even if the native architecture is different.
- 8 The `archnum` value in the Lead Section shall be `0x0001`.

Free Documentation License

2

3 **Free Documentation License**

Table of Contents

A. GNU Free Documentation License	1
A.1. PREAMBLE.....	1
A.2. APPLICABILITY AND DEFINITIONS.....	1
A.3. VERBATIM COPYING.....	2
A.4. COPYING IN QUANTITY.....	2
A.5. MODIFICATIONS.....	3
A.6. COMBINING DOCUMENTS.....	4
A.7. COLLECTIONS OF DOCUMENTS.....	4
A.8. AGGREGATION WITH INDEPENDENT WORKS.....	4
A.9. TRANSLATION.....	5
A.10. TERMINATION.....	5
A.11. FUTURE REVISIONS OF THIS LICENSE.....	5
A.12. How to use this License for your documents.....	5

Appendix A. GNU Free Documentation License

1 Version 1.1, March 2000

2 Copyright (C) 2000 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is
3 permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

A.1. PREAMBLE

4 The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to
5 assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or
6 noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work,
7 while not being considered responsible for modifications made by others.

8 This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the
9 same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

10 We have designed this License in order to use it for manuals for free software, because free software needs free
11 documentation: a free program should come with manuals providing the same freedoms that the software does. But
12 this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or
13 whether it is published as a printed book. We recommend this License principally for works whose purpose is
14 instruction or reference.

A.2. APPLICABILITY AND DEFINITIONS

15 This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be
16 distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member
17 of the public is a licensee, and is addressed as "you".

18 A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied
19 verbatim, or with modifications and/or translated into another language.

20 A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the
21 relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and
22 contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook
23 of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of
24 historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or
25 political position regarding them.

26 The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant
27 Sections, in the notice that says that the Document is released under this License.

28 The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the
29 notice that says that the Document is released under this License.

30 A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification
31 is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic
32 text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available
33 drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats
34 suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been

35 designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not
36 "Transparent" is called "Opaque".

37 Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format,
38 LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML
39 designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and
40 edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not
41 generally available, and the machine-generated HTML produced by some word processors for output purposes only.

42 The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold,
43 legibly, the material this License requires to appear in the title page. For works in formats which do not have any title
44 page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the
45 beginning of the body of the text.

A.3. VERBATIM COPYING

46 You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that
47 this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced
48 in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical
49 measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may
50 accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow
51 the conditions in section 3.

52 You may also lend copies, under the same conditions stated above, and you may publicly display copies.

A.4. COPYING IN QUANTITY

53 If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires
54 Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover
55 Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify
56 you as the publisher of these copies. The front cover must present the full title with all words of the title equally
57 prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the
58 covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim
59 copying in other respects.

60 If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit
61 reasonably) on the actual cover, and continue the rest onto adjacent pages.

62 If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a
63 machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a
64 publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of
65 added material, which the general network-using public has access to download anonymously at no charge using
66 public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you
67 begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the
68 stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents
69 or retailers) of that edition to the public.

70 It is requested, but not required, that you contact the authors of the Document well before redistributing any large
71 number of copies, to give them a chance to provide you with an updated version of the Document.

A.5. MODIFICATIONS

72 You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above,
73 provided that you release the Modified Version under precisely this License, with the Modified Version filling the role
74 of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of
75 it. In addition, you must do these things in the Modified Version:

- 76 A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of
77 previous versions (which should, if there were any, be listed in the History section of the Document). You may
78 use the same title as a previous version if the original publisher of that version gives permission.
 - 79 B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications
80 in the Modified Version, together with at least five of the principal authors of the Document (all of its principal
81 authors, if it has less than five).
 - 82 C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
 - 83 D. Preserve all the copyright notices of the Document.
 - 84 E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
 - 85 F. Include, immediately after the copyright notices, a license notice giving the public permission to use the
86 Modified Version under the terms of this License, in the form shown in the Addendum below.
 - 87 G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the
88 Document's license notice.
 - 89 H. Include an unaltered copy of this License.
 - 90 I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new
91 authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History"
92 in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title
93 Page, then add an item describing the Modified Version as stated in the previous sentence.
 - 94 J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the
95 Document, and likewise the network locations given in the Document for previous versions it was based on.
96 These may be placed in the "History" section. You may omit a network location for a work that was published at
97 least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
 - 98 K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the
99 section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
 - 100 L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or
101 the equivalent are not considered part of the section titles.
 - 102 M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
 - 103 N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.
- 104 If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and
105 contain no material copied from the Document, you may at your option designate some or all of these sections as
106 invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These
107 titles must be distinct from any other section titles.
- 108 You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified
109 Version by various parties--for example, statements of peer review or that the text has been approved by an
110 organization as the authoritative definition of a standard.

111 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
112 Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of
113 Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already
114 includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are
115 acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the
116 previous publisher that added the old one.

117 The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity
118 for or to assert or imply endorsement of any Modified Version.

A.6. COMBINING DOCUMENTS

119 You may combine the Document with other documents released under this License, under the terms defined in section
120 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the
121 original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

122 The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be
123 replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make
124 the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or
125 publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of
126 Invariant Sections in the license notice of the combined work.

127 In the combination, you must combine any sections entitled "History" in the various original documents, forming one
128 section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled
129 "Dedications". You must delete all sections entitled "Endorsements."

A.7. COLLECTIONS OF DOCUMENTS

130 You may make a collection consisting of the Document and other documents released under this License, and replace
131 the individual copies of this License in the various documents with a single copy that is included in the collection,
132 provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

133 You may extract a single document from such a collection, and distribute it individually under this License, provided
134 you insert a copy of this License into the extracted document, and follow this License in all other respects regarding
135 verbatim copying of that document.

A.8. AGGREGATION WITH INDEPENDENT WORKS

136 A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a
137 volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document,
138 provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and
139 this License does not apply to the other self-contained works thus compiled with the Document, on account of their
140 being thus compiled, if they are not themselves derivative works of the Document.

141 If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less
142 than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the
143 Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

A.9. TRANSLATION

144 Translation is considered a kind of modification, so you may distribute translations of the Document under the terms
145 of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders,
146 but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant
147 Sections. You may include a translation of this License provided that you also include the original English version of
148 this License. In case of a disagreement between the translation and the original English version of this License, the
149 original English version will prevail.

A.10. TERMINATION

150 You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License.
151 Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate
152 your rights under this License. However, parties who have received copies, or rights, from you under this License will
153 not have their licenses terminated so long as such parties remain in full compliance.

A.11. FUTURE REVISIONS OF THIS LICENSE

154 The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time
155 to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new
156 problems or concerns. See <http://www.gnu.org/copyleft/>.

157 Each version of the License is given a distinguishing version number. If the Document specifies that a particular
158 numbered version of this License "or any later version" applies to it, you have the option of following the terms and
159 conditions either of that specified version or of any later version that has been published (not as a draft) by the Free
160 Software Foundation. If the Document does not specify a version number of this License, you may choose any version
161 ever published (not as a draft) by the Free Software Foundation.

A.12. How to use this License for your documents

162 To use this License in a document you have written, include a copy of the License in the document and put the
163 following copyright and license notices just after the title page:

164 Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of
165 the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the
166 Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being
167 LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

168 If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you
169 have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for
170 Back-Cover Texts.

171 If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel
172 under your choice of free software license, such as the GNU General Public License, to permit their use in free
173 software.