ELC
PLATFORM SPECIFICATION
Version 1.0

Approved December 2002

The Embedded Linux Consortium

CONTENTS

1 Copyright and Licensing Terms

Embedded Linux Consortium Platform Specification v1.0

2 Acknowledgements

This specification was prepared by the Embedded Linux Consortium's Core Platform Specification Working Group, whose members were:

Mark Brown, IBM, Chair
Mitch Bunnell, Lynuxworks, Vice Chair
David A. Braun, Panasonic
Min Suk Choi, Samsung
Lee Courtney, MontaVista
Kevin Dankwardt, K Computing
Joe DeBlaquiere, Red Hat
Thiru Govindan, Wipro Technologies
Bao C. Ha, Hacom
Dr. I. P. Park, Panasonic
Greg Rose, Lynuxworks
Dongjun Shin, Samsung
Victor Yodaiken, FSM Labs

3 Introduction

This is Version 1.0 of the Embedded Linux Consortium Platform Specification (ELCPS). An implementation of this version of the specification may not claim to be an implementation of the ELCPS unless it has successfully completed the compliance process as defined by the Embedded Linux Consortium.

3.1 Purpose

The purpose of this specification is to define embedded system application programming environments (or profiles) based on the Linux operating system. This is intended for embedded system implementers and embedded application software developers. Embedded systems are systems either constrained or purposely optimized for a given environment.

This specification is built upon a much larger and widely supported set of standards, in particular:

- The Linux Standards Base 1.2.
- The IEEE POSIX 1003.1-2001 specification, which supersedes the 1996 version and contains updates for Realtime, Threads and Networking.
- The Single UNIX Specification v3, which supersedes the UNIX98 standard and was produced in conjunction with IEEE POSIX 1003.1-2001.

These allow for the formation of a specification with a sound footing in industry-standard behavior. At the same time, this document is designed to allow for extension and future enhancement as the industry progresses.

This standard defines three environments to reflect the wide range of system requirements presented by embedded designs. The intent is to provide meaningful and coherent sets of interfaces that will present software vendors and consumers with a uniform framework for describing and specifying system capabilities. This allows an application writer to construct an application that may be easily moved to a different system that supports the same environment. Similarly, it allows a vendor to claim conformance with an established specification.

This specification is designed to support the common practice of interconnecting several smaller systems to create larger systems. Each interconnected system may use different ELCPS (or other) environments. For example, one can envision a hierarchical system where the bottom-level elements (e.g., device controllers) use the "minimal" environment, the next level up uses the somewhat larger "intermediate" environment, and so on. For this reason the Platform Specification specifies interfaces for the smaller environments that make no sense for an isolated system. These interfaces are specified to support the construction of hierarchical systems as well as systems of communicating heterogeneous peers.

In summary, the ELCPS aims:

- To promote development of embedded Linux systems and applications,
- To allow for scalability in those environments, based on intended uses,
- To promote portability of embedded Linux applications,

and it will do this by

- Using existing Linux and UNIX industry standards
- Allowing for adaptation to existing Linux common practice
- Breaking down the environments into recognized sets of function, for configurability.

3.2 Relationship to Other Industry Standards

The specifications listed below are referenced in whole or in part by the ELCPS. Such references may be normative or non-normative[1]; a reference to specification shall only be considered normative if it is explicitly cited as such. The ELCPS makes normative references to portions of:

ISOC99	ISO/IEC 9899:1999, Programming Languages - C		
LSB1.2	Linux Standard Base	http://www.linuxbase.org/spec/	1
POSIX.1-2001	IEEE POSIX 1003.1-2001	http://www.ieee.org	2
SUSV3	Open Group Single UNIX Specification version 3	http://www.opengroup.org	2,3

Notes:
1. This document is available without charge at the URL cited.
2. These documents are actually the same document, containing different sections for the appropriate standard. ISO is also intending to affirm this document as a superseding standard to ISO/IEC 9945-1:1996. The goal was to get rid of conflicts and omissions between the various standards.
3. This document (the same text as POSIX.1-2001 under the SUS title) is publicly available without charge at the URL cited. You will need to register to obtain a copy at this time.

Any conflict between this specification and any of these standards is unintentional. This document defers to the formal standards, which the ELCPS recognizes as superior, unless explicitly excepted in the specification. In particular, from time to time, when ambiguities or discrepancies are found in the formal standards, the responsible bodies will make interpretations of them, whose findings will become binding on this Specification. Where, as the result of such an interpretation, or for any other reason, any of these formal standards are found to conflict with this specification (and such conflict is not explicitly excepted in the specification), ELCPS-conformant systems may offer behavior defined by the formal standards or by this specification. ELCPS-conformant systems must document which behavior they offer. Application writers should avoid depending exclusively on either behavior in such cases.

[1] "Normative" text in a specification document is that text that is part of the formal specification. Its counterpart is "Informative" text, which may add to the information in the specification but is not an official part of the specification itself.

3.3 How To Use This Specification

The general approach taken in this specification is to create functional groups of system interfaces, taken from the LSB, POSIX, and the SUSv3 sufficient to deliver the functionality typical of current embedded Linux systems. Each environment is specified with full features, to give users clear direction. Implementers must provide all required features for an environment, but may provide means to configure out those parts not needed by a specific application.

Implementers wishing to expand on the specified environments are strongly encouraged to take the added interfaces from current Linux practice or from the base standards, rather than invent new interfaces.

For each profile, the minimum hardware typically required is specified. This is the hardware assumed to be present; implementations may of course have more, but nothing in the profile requires - either directly or indirectly - more than the specified minimum hardware model.

This document should be used in conjunction with the documents it references. This document enumerates the system elements and interfaces it includes, but descriptions and specifications of those elements and interfaces may be included entirely or partly in this document, or entirely in other referenced documents. For example, the section that describes interface groupings includes a list of the system APIs supported in each group, and a pointer to the underlying referenced specification for information about the syntax and semantics of each interface. Only those routines not described in standards referenced by this document, or extensions to those standards, are described in this specification itself. Information referenced in this way is as much a part of this document as is the information explicitly included here.

3.4 Definitions

3.4.1 ELCPS

This document.

3.4.2 ELCPS-Compliant Application

An application written to reference or invoke only the system APIs and other resources specified in this document.

3.4.3 ELCPS-Conforming Implementation

An implementation that provides the system environment(s) for applications as described in this document, and has successfully completed the requirements for claiming conformance, as defined by the ELC.

3.4.4 Non-ELCPS-Compliant Application

An application which has been written to reference or invoke system routines, commands, or other resources not specified in this document.

3.4.5 ELCPS Implementation Conformance

An implementation satisfying the following requirements:

- The implementation shall provide the interface function groups specified by this document for a given environment.
- The implementation shall provide all of the mandatory interface function groups for a given environment, in their entirety.
- The implementation may provide one or more of the non-mandatory interface function groups in a given environment. The optional groups for which conformance is claimed, shall be provided in their entirety. The product documentation shall state which optional interface groups are provided.

The implementation may provide additional interfaces with different names. It may also provide additional behavior corresponding to data values outside the standard ranges, for standard named interfaces.

3.4.6 ELCPS Application Conformance

An application with the following characteristics:

- If it requires any optional interface defined in this document in order to be installed or to execute successfully, the requirement for that optional interface is stated in the application's documentation.
- It does not use any interface or data format that is not required to be provided by a conforming implementation, unless:
- If such an interface or data format is supplied by another application through direct invocation of that application during execution, that application is in turn an ELCPS-compliant application.
- The use of that interface or data format, as well as its source, is identified in the documentation of the application.
- It must not use any values for a named interface that are reserved for vendor extensions.

3.4.7 ELCPS Strictly Conforming Application

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

3.5 Terminology

3.5.1 can

Describes a permissible feature or behavior available to the user or application. The feature or behavior is mandatory for an implementation that conforms to this document. An application can rely on the existence of the feature or behavior.

3.5.2 implementation-defined

(Same meaning as implementation-dependent.) Describes a value or behavior that is not defined by this document but is selected by an implementer. The value or behavior is allowed to vary among implementations that conform to this document. An application should not rely on the existence of the value or behavior. An application that relies on such a value or behavior cannot be assured to be portable across conforming implementations. The implementer shall document such a value or behavior so that it can be used correctly by an application.

3.5.3 may

Describes a feature or behavior that is optional for an implementation that conforms to this document. An application should not rely on the existence of the feature or behavior. An application that relies on such a feature or behavior cannot be assured to be portable across conforming implementations. To avoid ambiguity, the opposite of may is expressed as need not, instead of may not.

3.5.4 must

Describes a feature or behavior that is mandatory for an application or user. An implementation that conforms to this document shall support this feature or behavior.

3.5.5 shall

Describes a feature or behavior that is mandatory for an implementation that conforms to this document. An application can rely on the existence of the feature or behavior.

3.5.6 should

For an implementation that conforms to this document, describes a feature or behavior that is recommended but not mandatory. An application should not rely on the existence of the feature or behavior. An application that relies on such a feature or behavior cannot be assured to be portable across conforming implementations.

For an application, describes a feature or behavior that is recommended programming practice for optimum portability.

3.5.7 undefined

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

3.5.8 unspecified

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

4 System Environments

This section defines a set of "system environments for applications" for embedded Linux systems, beginning with a minimal environment and adding groups of function as the environments grow larger and more complex. The organization and makeup of these environments is heavily influenced by the IEEE POSIX 1003.13 "Standardized Application Environment Profile - POSIX Realtime Application Support (AEP)". While this first version of the ELCPS does not directly address RTOS issues, many of the basic principles stated in 1003.13 are the same.

These environments are designed such that it is possible to provide each of them from a fully conforming LSB1.2 system implementation. Each environment is purposely designed to be a proper subset of the next larger environment.

4.1 Minimal System Environment

This environment describes systems that are typically deeply embedded and dedicated to isolated/unattended operation of one or more special devices. They require minimal or no user interaction, and may not require such features as mass storage (such as a file system). There is usually only one actual process, possibly with one or more threads of control (Linux tasks or POSIX threads). There may be multiple processes using only one address space (the POSIX *fork()* API may not be available).

The only hardware assumed in this environment is a single processor with its memory.

4.2 Intermediate System Environment

This takes the Minimal Environment and adds support for mass storage (file and file system interfaces, including Linux Large File Support), Asynchronous (non-blocking) I/O, dynamic linking of objects (libraries). Multiple processes or address spaces are possible.

The hardware requirements do not assume actual mass storage, the filesystem may be implemented by other means, such as RAM or ROM. One or more processors with associated memory are assumed.

4.3 Full System Environment

This is essentially a full, multi-purpose Linux environment, including all of the function of the other, smaller environments. This is essentially equivalent to a LSB1.2 system, with the exception that no actual system utilities are specified (but the POSIX shell is indeed specified in this environment via functions such as popen()).

The hardware model includes one or more processors with memory, mass storage, network support and user interface/display devices.

5 Environment Function Group Tables

5.1 Required Environment Function Groups

The following table represents the API function groups, and their status for each of the System Environments[2]:

R - Required for this Environment
P - Optional for this Environment, but required for POSIX conformance.
L - Optional for this Environment, but required for LSB1.2 conformance.

In this table, all the entries with no label (R, P, or L) are optional, and can be offered in a given environment but are not mandatory for that environment. Environments with P/L entries must offer at least one, and may offer both. Implementations must document if they are offering P, L, or both. If both are offered, the use and interaction of the two in the environment must be documented.

Implementations must document which optional groups, if any, are provided in an environment.

	Minimal SE	**Intermediate SE**	**Full SE**
ELC_ASYNCHRONOUS_IO		R	R
ELC_C_LANG_JUMP		R	R
ELC_C_LANG_MATH			R
ELC_C_LANG_SUPPORT	R	R	R
ELC_C_LANG_SUPPORT_R	R	R	R
ELC_C_LIB_EXT		R	R
ELC_DEVICE_IO		R	R
ELC_DEVICE_SPECIFIC			R
ELC_DEVICE_SPECIFIC_R			R
ELC_DYNAMIC_LINKING		R	R
ELC_FD_MGMT		R	R
ELC_FIFO			R
ELC_FILE_ATTRIBUTES			R
ELC_FILE_SYSTEM		R	R
ELC_FILE_SYSTEM_EXT			R
ELC_FILE_SYSTEM_R		R	R
ELC_IPC		R	R
ELC_JOB_CONTROL			R
ELC_JUMP		R	R
ELC_LARGE_FILE		R	R
ELC_LSB_THREADS	L	L	L
ELC_LSB_THREADS_EXT		L	L

[2] The term “Environment” is used here in the same way that “Profile” is used in IEEE POSIX specifications.

ELC_MEM_MGMT		R	R
ELC_MULTI_ADDR_SPACE		R	R
ELC_MULTI_PROCESS		R	R
ELC_NETWORKING			R
ELC_NETWORKING_RPC			R
ELC_PIPE		R	R
ELC_POSIX_THREADS	P	P	P
ELC_POSIX_THREADS_EXT		P	P
ELC_REGEXP			R
ELC_SHELL_FUNC			R
ELC_SIGNALS	R	R	R
ELC_SIGNAL_JUMP		R	R
ELC_SINGLE_PROCESS	R	R	R
ELC_STDIO_LOCKING	R	R	R
ELC_SYMBOLIC_LINKS			R
ELC_SYSTEM_DATABASE			R
ELC_SYSTEM_DATABASE_R			R
ELC_SYSTEM_LOGGING			R
ELC_USER_GROUPS			R
ELC_USER_GROUPS_R			R
ELC_WIDE_CHAR		R	R
ELC_WIDE_CHAR_DEVICE_IO		R	R

5.2 POSIX 1003.1-2001 Feature Options

The following table represents the POSIX 1003.1-2001 Feature Options, and their status for each of the System Environments. The POSIX Feature Options below are functions that are optional as to base POSIX 1003.1-2001 conformance requirements, but useful in embedded OS environments.

R - required for this Environment

	Minimal SE	Intermediate SE	Full SE
NGROUPS_MAX			>=8
_POSIX_CHOWN_RESTRICTED			R
_POSIX_FSYNC	R	R	R
_POSIX_JOB_CONTROL			R
_POSIX_MESSAGE_PASSING	R	R	R
_POSIX_NO_TRUNC	R	R	R
_POSIX_REGEXP			R
_POSIX_READER_WRITER_LOCKS	R	R	R
_POSIX_SAVED_IDS			R
_POSIX_VDISABLE			R

6 Interface Function Groups

The following sections represent the groupings of APIs into areas of function. These groupings are used in the ELCPS to represent what function is required at each level of conformance. Each group's elements will be separated to indicate the specification upon which they are based:

- POSIX.1-2001 is a reference to IEEE POSIX 1003.1-2001, including Rationale
- LSB1.2 is a reference to Linux Standard Base Version 1.2.0
- SUSv3 is a reference to the Single UNIX Specification, Version 3

All interfaces included in any one of the function groups below, shall behave as described and defined in the normative parts of the referenced standard containing them.

6.1 Threads

The ELCPS offers two different versions of thread APIs: LSB1.2-based and POSIX-based. An implementation must support at least one of the two, and may choose to support both.

Applications should be written to deal with either form of threads support. An implementation choosing to support both models and multiple applications, must allow for applications individually choosing which model to use. Sets of cooperating applications must agree on a common threads model to use.

Linux historically has supported the POSIX threads (pthreads) API set, but differed in underlying organization and semantics. The LSB1.2-based groups are included to reflect this historic behavior.

6.2 Realtime

While the purpose of this document is to specify embedded Linux system environments, one set of function (Asynchronous I/O) from the Realtime Options of POSIX.1-2001 has been included in this specification.

6.3 Listing of Function Groups

Some APIs may be present in more than one function group. This reflects the fact that some interfaces have purposes valid for more than one grouping, and that some interfaces may have different required behaviors when certain optional features such as threads are active.

6.3.1 ELC_ASYNCHRONOUS_IO

(Asynchronous I/O) contains:
The set of APIs described in the POSIX.1-2001 Feature Group _POSIX_ASYNCHRONOUS_IO:

aio_cancel(), aio_error(), aio_fsync(), aio_read(), aio_return(), aio_suspend(), aio_write(), aio_listio(),

The following APIs as defined in LSB1.2:

aio_cancel64(), aio_error64(), aio_fsync64(), aio_read64(), aio_return64(), aio_suspend64(), aio_write64(), lio_listio64(),

With the exception of the following APIs, which are excluded from this set: None

6.3.2 ELC_C_LANG_JUMP

(ISO C Library Jump Functions) contains
The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_C_LANG_JUMP:

longjmp(), setjmp()

The following APIs as defined in LSB1.2: None
With the exception of the following APIs, which are excluded from this set: None

6.3.3 ELC_C_LANG_MATH

(Math Functions) contains
The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_C_LANG_MATH:

acos(), acosf(), acosh(), acoshf(), acoshl(), acosl(), asin(), asinf(), asinh(), asinhf(), asinhl(), asinl(), atan(), atan2(), atan2f(), atan2l(), atanf(), atanh(), atanhf(), atanhl(), atanl(), cabs(), cabsf(), cabsl(), cacos(), cacosf(), cacosh(), cacoshf(), cacoshl(), cacosl(), carg(), cargf(), cargl(), casin(), casinf(), casinh(), casinhf(), casinhl(), casinl(), catan(), catanf(), catanh(), catanhf(), catanhl(), catanl(), cbrt(), cbrtf(), cbrtl(), ccos(), ccosf(), ccosh(), ccoshf(), ccoshl(), ccosl(), ceil(), ceilf(), ceill(), cexp(), cexpf(), cexpl(), cimag(), cimagf(), cimagl(), clog(), clogf(), clogl(), conj(), conjf(), conjl(), copysign(), copysignf(), copysignl(), cos(), cosf(), cosh(), coshf(), coshl(), cosl(), cpow(), cpowf(), cpowl(), cproj(), cprojf(), cprojl(), creal(), crealf(), creall(), csin(), csinf(), csinh(), csinhf(), csinhl(), csinl(), csqrt(), csqrtf(), csqrtl(), ctan(), ctanf(), ctanh(), ctanhf(), ctanhl(), ctanl(), erf(), erfc(), erfcf(), erfcl(), erff(), erfl(), exp(), exp2(), exp2f(), exp2l(), expf(), expl(), expm1(), expm1f(), expm1l(), fabs(), fabsf(), fabsl(), fdim(), fdimf(), fdiml(), floor(), floorf(), floorl(), fma(), fmaf(), fmal(), fmax(), fmaxf(), fmaxl(), fmin(), fminf(), fminl(), fmod(), fmodf(), fmodl(), fpclassify(), frexp(), frexpf(), frexpl(), hypot(), hypotf(), hypotl(), ilogb(), ilogbf(), ilogbl(), isfinite(), isgreater(), isgreaterequal(), isinf(), isless(), islessequal(), islessgreater(), isnan(), isnormal(), isunordered(), ldexp(), ldexpf(), ldexpl(), lgamma(), lgammaf(), lgammal(), llrint(), llrintf(), llrintl(), llround(), llroundf(), llroundl(), log(), log10(), log10f(), log10l(), log1p(), log1pf(), log1pl(), log2(), log2f(), log2l(), logb(), logbf(), logbl(), logf(), logl(), lrint(), lrintf(), lrintl(), lround(), lroundf(), lroundl(), modf(), modff(), modfl(), nan(), nanf(), nanl(), nearbyint(), nearbyintf(), nearbyintl(), nextafter(), nextafterf(), nextafterl(), nexttoward(), nexttowardf(), nexttowardl(), pow(), powf(), powl(), remainder(), remainderf(), remainderl(), remquo(), remquof(), remquol(), rint(), rintf(), rintl(), round(), roundf(), roundl(), scalbln(), scalblnf(), scalblnl(), scalbn(), scalbnf(), scalbnl(), signbit(), sin(), sinf(), sinh(), sinhf(), *sinhl(), sinl(), sqrt(), sqrtf(), sqrtl(), tan(), tanf(), tanh(), tanhf(), tanhl(), tanl(), tgamma(), tgammaf(), tgammal(), trunc(), truncf(), truncl()*

The set of APIs described in SUSv3 Appendix E.1, XSI_MATH:

j0(), j1(), jn(), scalb(), y0(), y1(), yn()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.4 ELC_C_LANG_SUPPORT

(General ISO C Library) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_C_LANG_SUPPORT:

abs(), asctime(), atof(), atoi(), atol(), atoll(), bsearch(), calloc(), ctime(), difftime(), div(), feclearexcept(), fegetenv(), fegetexceptflag(), fegetround(), feholdexcept(), feraiseexcept(), fesetenv(), fesetexceptflag(), fesetround(), fetestexcept(), feupdateenv(), free(), gmtime(), imaxabs(), imaxdiv(), isalnum(), isalpha(), isblank(), iscntrl(), isdigit(), isgraph(), islower(), isprint(), ispunct(), isspace(), isupper(), isxdigit(), labs(), ldiv(), llabs(), lldiv(), localeconv(), localtime(), malloc(), memchr(), memcmp(), memcpy(), memmove(), memset(), mktime(), qsort(), rand(), realloc(), setlocale(), snprintf(), sprintf(), srand(), sscanf(), strcat(), strchr(), strcmp(), strcoll(), strcpy(), strcspn(), strerror(), strftime(), strlen(), strncat(), strncmp(), strncpy(), strpbrk(), strrchr(), strspn(), strstr(), strtod(), strtof(), strtoimax(), strtok(), strtol(), strtold(), strtoll(), strtoul(), strtoull(), strtoumax(), strxfrm(), time(), tolower(), toupper(), tzname, tzset(), va_arg(), va_copy(), va_end(), va_start(), vsnprintf(), vsprintf(), vsscanf()

The set of APIs described in SUSv3 Appendix E.1, XSI_C_LANG_SUPPORT*:*

_tolower(), _toupper(), a64l(), daylight(), drand48(), erand48(), ffs(), getcontext(), getdate(), getsubopt(), hcreate(), hdestroy(), hsearch(), iconv(), iconv_close(), iconv_open(), initstate(), insque(), isascii(), jrand48(), l64a(), lcong48(), lfind(), lrand48(), lsearch(), makecontext(), memccpy(), mrand48(), nrand48(), random(), remque(), seed48(), setcontext(), setstate(), signgam, srand48(), srandom(), strcasecmp(), strdup(), strfmon(), strncasecmp(), strptime(), swab(), swapcontext(), tdelete(), tfind(), timezone(), toascii(), tsearch(), twalk()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.5 ELC_C_LANG_SUPPORT_R

(Thread-Safe General ISO C Library) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_C_LANG_SUPPORT_R:

asctime_r(), ctime_r(), gmtime_r(), localtime_r(), rand_r(), strerror_r(), strtok_r()

The following APIs as defined in LSB1.2:

random_r(),

With the exception of the following APIs, which are excluded from this set: None

6.3.6 ELC_C_LIB_EXT

(General C Library Extension) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_C_LIB_EXT:

fnmatch(), getopt(), optarg, opterr, optind, optopt

The following APIs as defined in LSB1.2:

stime(), getopt_long(), memmem(), getopt_long_only(), memrchr(), stpcpy(), stpncpy(), strcasestr(), strndup(), strnlen(), strsep(), strsignal(), strtoq(), strtouq(), strverscmp(), adjtime(), adjtimex(),

With the exception of the following APIs, which are excluded from this set:

brk() [see 6.3.24 ELC_MULTI_ADDR_SPACE]

6.3.7 ELC_DEVICE_IO

(Device Input and Output) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_DEVICE_IO:

FD_CLR(), FD_ISSET(), FD_SET(), FD_ZERO(), clearerr(), close(), fclose(), fdopen(), feof(), ferror(), fflush(), fgetc(), fgets(), fileno(), fopen(), fprintf(), fputc(), fputs(), fread(), freopen(), fscanf(), fwrite(), getc(), getchar(), gets(), open(), perror(), printf(), pselect(), putc(), putchar(), puts(), read(), scanf(), select(), setbuf(), setvbuf(), stderr, stdin, stdout, ungetc(), vfprintf(), vfscanf(), vprintf(), vscanf(), write()

The set of APIs described in SUSv3 Appendix E.1, XSI_DEVICE_IO:

fmtmsg(), poll(), pread(), pwrite(), readv(), writev()

The following APIs as defined in LSB1.2:

vasprintf(), vdprintf(), setbuffer(), err(), error(), errx(), verrx(), warn(), warnx(),

With the exception of the following APIs, which are excluded from this set: None

6.3.8 ELC_DEVICE_SPECIFIC

(General Terminal) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_DEVICE_SPECIFIC:

cfgetispeed(), cfgetospeed(), cfsetispeed(), cfsetospeed(), ctermid(), isatty(), tcdrain(), tcflow(), tcflush(), tcgetattr(), tcsendbreak(), tcsetattr(), ttyname()

The set of APIs described in SUSv3 Appendix E.1, XSI_DEVICE_SPECIFIC:

grantpt(), posix_openpt(), ptsname(), unlockpt()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.9 ELC_DEVICE_SPECIFIC_R

(Thread-Safe General Terminal) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_DEVICE_SPECIFIC_R:

ttyname_r()

The following APIs as defined in LSB1.2:

cfmakeraw(), cfsetspeed(),

With the exception of the following APIs, which are excluded from this set: None

6.3.10 ELC_DYNAMIC_LINKING

(Dynamic Linking) contains

The set of APIs described in SUSv3 Appendix E.1, XSI_DYNAMIC_LINKING:

dlclose(), dlerror(), dlopen(), dlsym()

The following APIs as defined in LSB1.2:

dladdr(),

With the exception of the following APIs, which are excluded from this set: None

6.3.11 ELC_FD_MGMT

(File Descriptor Management) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_FD_MGMT:

dup(), dup2(), fcntl(), fgetpos(), fseek(), fseeko(), fsetpos(), ftell(), ftello(), ftruncate(), lseek(), rewind()

The set of APIs described in SUSv3 Appendix E.1, XSI_FD_MGMT:

truncate()

The following APIs as defined in LSB1.2:

flock()

With the exception of the following APIs, which are excluded from this set: None

6.3.12 ELC_FIFO

(FIFO) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_FIFO:

mkfifo()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.13 ELC_FILE_ATTRIBUTES

(File Attributes) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_FILE_ATTRIBUTES:

chmod(), chown(), fchmod(), fchown(), umask()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.14 ELC_FILE_SYSTEM

(File System) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_FILE_SYSTEM:

access(), chdir(), closedir(), creat(), fpathconf(), fstat(), getcwd(), link(), mkdir(), opendir(), pathconf(), readdir(), remove(), rename(), rewinddir(), rmdir(), stat(), tmpfile(), tmpnam(), unlink(), utime()

The set of APIs described in SUSv3 Appendix E.1, XSI_FILE_SYSTEM:

basename(), dirname(), fchdir(), fstatvfs(), ftw(), lchown(), lockf(), mknod(), mkstemp(), nftw(), realpath(), seekdir(), statvfs(), sync(), telldir(), tempnam()

The following APIs as defined in LSB1.2:

alphasort( ), statfs(), fstatfs(),

With the exception of the following APIs, which are excluded from this set: None

6.3.15 ELC_FILE_SYSTEM_EXT

(File System Extensions) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_FILE_SYSTEM_EXT:

glob(), globfree()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.16 ELC_FILE_SYSTEM_R

(Thread-Safe File System) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_FILE_SYSTEM_R:

readdir_r()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.17 ELC_IPC

(Interprocess Communication) contains

The set of APIs described in SUSv3 Appendix E.1, XSI_IPC:

ftok(), msgctl(), msgget(), msgrcv(), msgsnd(), semctl(), semget(), semop(), shmat(), shmctl(), shmdt(), shmget()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.18 ELC_JOB_CONTROL

(Job Control) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_JOB_CONTROL:

setpgid(), tcgetpgrp(), tcsetpgrp()

The setof APIs described in SUSv3 Appendix E.1, XSI_JOB_CONTROL:

tcgetsid()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.19 ELC_JUMP

(Extended Jump Functions) contains

The set of APIs described in SUSv3 Appendix E.1, XSI_JUMP:

_longjmp(), _setjmp()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.20 ELC_LARGE_FILE

(Large File Support) contains

The following APIs as defined in LSB1.2:

globfree64(), glob64(), fopen64(), ftello64(), mkstemp64(), tmpfile64(), freopen64(), ftruncate64(), mmap64(), truncate64(), fseeko64(), ftw64(), nftw64(), alphasort64(), fsetpos64(), getrlimit64(), open64(), creat64(), fstatfs64(), lockf64(), pwrite64(), fgetpos64(), fstatvfs64(), lseek64(), readdir64(),

6.3.21 ELC_LSB_THREADS

(LSB-conforming threads) contains

The set of APIs described in POSIX.1-2001 Option Groups: _POSIX_THREADS, _POSIX_THREAD_ATTR_STACKADDR, _POSIX_THREAD_ATTR_STACKSIZE, _POSIX_READER_WRITER_LOCKS, _POSIX_THREAD_SAFE_FUNCTIONS:

pthread_atfork(), pthread_attr_destroy(), pthread_attr_getdetachstate(), pthread_attr_getguardsize(), pthread_attr_getschedparam(), pthread_attr_getstack(), pthread_attr_getstackaddr(), pthread_attr_getstacksize(), pthread_attr_init(), pthread_attr_setdetachstate(), pthread_attr_setguardsize(), pthread_attr_setschedparam(), pthread_attr_setstack(), pthread_attr_setstackaddr(), pthread_attr_setstacksize(), pthread_cancel(), pthread_cleanup_pop(), pthread_cleanup_push(), pthread_cond_broadcast(), pthread_cond_destroy(), pthread_cond_init(), pthread_cond_signal(), pthread_cond_timedwait(), pthread_cond_wait(), pthread_condattr_destroy(), pthread_key_create(), pthread_key_delete(), pthread_kill(), pthread_mutex_destroy(), pthread_mutex_init(), pthread_mutex_lock(), pthread_mutex_trylock(), pthread_mutex_unlock(), pthread_mutexattr_destroy(), pthread_mutexattr_gettype(), pthread_mutexattr_init(), pthread_mutexattr_settype(), pthread_once(), pthread_rwlock_destroy(), pthread_rwlock_init(), pthread_rwlock_rdlock(), pthread_rwlock_tryrdlock(), pthread_rwlock_trywrlock(), pthread_rwlock_unlock(), pthread_rwlock_wrlock(), pthread_rwlockattr_destroy(), pthread_rwlockattr_init(), pthread_self(), pthread_setcancelstate(), pthread_setcanceltype(), pthread_setconcurrency(), pthread_setspecific(), pthread_sigmask(), pthread_testcancel(), sigwait(), pthread_condattr_init(), pthread_create(), pthread_detach(), pthread_equal(), pthread_exit(), pthread_getconcurrency(), pthread_getspecific(), pthread_join(), asctime_r(), ctime_r(), flockfile(), ftrylockfile(), funlockfile(), getc_unlocked(), getchar_unlocked(), getgrgid_r(), getgrnam_r(), getpwnam_r(), getpwuid_r(), gmtime_r(), localtime_r(), putc_unlocked(), putchar_unlocked(), rand_r(), readdir_r(), strerror_r(), strtok_r()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

All APIs in this group behave as defined in LSB1.2.

6.3.22 ELC_LSB_THREADS_EXT

(LSB-threads extensions) contains

The set of APIs described in POSIX.1-2001 Option Groups: _POSIX_THREAD_PROCESS_SHARED:

pthread_mutexattr_getpshared(), pthread_mutexattr_setpshared(), pthread_rwlockattr_getpshared(), pthread_rwlockattr_setpshared(), pthread_condattr_getpshared(), pthread_condattr_setpshared()

The set of APIs described in SUSv3 Appendix E.1: XSI_THREAD_MUTEX_EXT, XSI_THREADS_EXT:

pthread_mutexattr_gettype(), pthread_mutexattr_settype()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

All APIs in this group behave as defined in LSB1.2.

6.3.23 ELC_MEM_MGMT

(Memory Management) contains

The set of APIs described in POSIX.1-2001 Option Groups: _POSIX_MAPPED_FILES, _POSIX_MEMORY_PROTECTION, _POSIX_MEMLOCK, _POSIX_MEMLOCK_RANGE:

mmap(), mprotect(), msync(), munmap()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.24 ELC_MULTI_ADDR_SPACE

(Multiple Address Spaces) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_MULTI_PROCESS:

fork()

The set of APIs described in SUSv3 Appendix E.1: None

The following APIs as defined in LSB1.2:

brk()

With the exception of the following APIs, which are excluded from this set: None

6.3.25 ELC_MULTI_PROCESS

(Multiple Processes) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_MULTI_PROCESS:

_Exit(), _exit(), assert(), atexit(), clock(), execl(), execle(), execlp(), execv(), execve(), execvp(), exit(), getpgrp(), getpid(), getppid(), setsid(), sleep(), times(), wait(), waitpid()

The set of APIs described in SUSv3 Appendix E.1, XSI_MULTI_PROCESS:

getpgid(), getpriority(), getrlimit(), getrusage(), getsid(), nice(), setpgrp(), setpriority(), setrlimit(), ulimit(), usleep(), vfork(), waitid()

The following APIs as defined in LSB1.2:

wait4(), getloadavg(), daemon(),

With the exception of the following APIs, which are excluded from this set:

fork() [see 6.3.24 ELC_MULTI_ADDR_SPACE]

6.3.26 ELC_NETWORKING

(Networking) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_NETWORKING:

accept(), bind(), connect(), endhostent(), endnetent(), endprotoent(), endservent(), freeaddrinfo(), gai_strerror(), getaddrinfo(), gethostbyaddr(), gethostbyname(), gethostent(), gethostname(), getnameinfo(), getnetbyaddr(), getnetbyname(), getnetent(), getpeername(), getprotobyname(), getprotobynumber(), getprotoent(), getservbyname(), getservbyport(), getservent(), getsockname(), getsockopt(), h_errno, htonl(), htons(), if_freenameindex(), if_indextoname(), if_nameindex(), if_nametoindex(), inet_addr(), inet_ntoa(), inet_ntop(), inet_pton(), listen(), ntohl(), ntohs(), recv(), recvfrom(), recvmsg(), send(), sendmsg(), sendto(), sethostent(), setnetent(), setprotoent(), setservent(), setsockopt(), shutdown(), socket(), sockatmark(), socketpair()

The following APIs as defined in LSB1.2:

sethostname(), sethostid(), bindresvport(), gethostbyname_r(),

With the exception of the following APIs, which are excluded from this set: None

6.3.27 ELC_NETWORKING_RPC

(RPC) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_NETWORKING: None

The following APIs as defined in LSB1.2:

authnone_create(), clnt_create(), clnt_pcreateerror(), clnt_perrno(), clnt_perror(), clnt_spcreateerror(), clnt_sperrno(), clnt_sperror(), key_decryptsession(), svc_getreqset(), svcerr_auth(), svcerr_decode(), svcerr_noproc(), svcerr_noprog(), svcerr_progvers(), svcerr_systemerr(), svcerr_weakauth(), xdr_accepted_reply(), xdr_array(), xdr_bool(), xdr_bytes(), xdr_callhdr(), xdr_callmsg(), xdr_char(), xdr_double(), xdr_enum(), xdr_float(), xdr_free(), xdr_int(), xdr_long(), xdr_opaque(), xdr_opaque_auth(), xdr_pointer(), xdr_reference(), xdr_rejected_reply(), xdr_replymsg(), xdr_short(), xdr_string(), xdr_u_char(), xdr_u_long(), xdr_u_short(), xdr_union(), xdr_vector(), xdr_void(), xdr_wrapstring(), xdrmem_create(), xdrrec_create(), xdrrec_eof(),

With the exception of the following APIs, which are excluded from this set: None

6.3.28 ELC_PIPE

(Pipe) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_PIPE:

pipe()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.29 ELC_POSIX_THREADS

(POSIX-conforming threads) contains

The set of APIs described in POSIX.1-2001 Option Groups: _POSIX_THREADS, _POSIX_THREAD_ATTR_STACKADDR, _POSIX_THREAD_ATTR_STACKSIZE, _POSIX_READER_WRITER_LOCKS, _POSIX_THREAD_SAFE_FUNCTIONS:

pthread_atfork(), pthread_attr_destroy(), pthread_attr_getdetachstate(), pthread_attr_getguardsize(), pthread_attr_getschedparam(), pthread_attr_getstack(), pthread_attr_getstackaddr(), pthread_attr_getstacksize(), pthread_attr_init(), pthread_attr_setdetachstate(), pthread_attr_setguardsize(), pthread_attr_setschedparam(), pthread_attr_setstack(), pthread_attr_setstackaddr(), pthread_attr_setstacksize(), pthread_cancel(), pthread_cleanup_pop(), pthread_cleanup_push(), pthread_cond_broadcast(), pthread_cond_destroy(), pthread_cond_init(), pthread_cond_signal(), pthread_cond_timedwait(), pthread_cond_wait(), pthread_condattr_destroy(), pthread_key_create(), pthread_key_delete(), pthread_kill(), pthread_mutex_destroy(), pthread_mutex_init(), pthread_mutex_lock(), pthread_mutex_trylock(), pthread_mutex_unlock(), pthread_mutexattr_destroy(), pthread_mutexattr_gettype(), pthread_mutexattr_init(), pthread_mutexattr_settype(), pthread_once(), pthread_rwlock_destroy(), pthread_rwlock_init(), pthread_rwlock_rdlock(), pthread_rwlock_tryrdlock(), pthread_rwlock_trywrlock(), pthread_rwlock_unlock(), pthread_rwlock_wrlock(), pthread_rwlockattr_destroy(), pthread_rwlockattr_init(), pthread_self(), pthread_setcancelstate(), pthread_setcanceltype(), pthread_setconcurrency(), pthread_setspecific(), pthread_sigmask(), pthread_testcancel(), sigwait(), *pthread_condattr_init(), pthread_create(), pthread_detach(), pthread_equal(), pthread_exit(), pthread_getconcurrency(), pthread_getspecific(), pthread_join(), asctime_r(), ctime_r(), flockfile(), ftrylockfile(), funlockfile(), getc_unlocked(), getchar_unlocked(), getgrgid_r(), getgrnam_r(), getpwnam_r(), getpwuid_r(), gmtime_r(), localtime_r(), putc_unlocked(), putchar_unlocked(), rand_r(), readdir_r(), strerror_r(), strtok_r()*

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

All APIs in this group behave as defined in POSIX.1-2001.

6.3.30 ELC_POSIX_THREADS_EXT

(POSIX-threads extensions) contains

The set of APIs described in POSIX.1-2001 Option Groups:

_POSIX_THREAD_PROCESS_SHARED:

pthread_mutexattr_getpshared(), pthread_mutexattr_setpshared(), pthread_rwlockattr_getpshared(), pthread_rwlockattr_setpshared(), pthread_condattr_getpshared(), pthread_condattr_setpshared()

The set of APIs described in SUSv3 Appendix E.1: XSI_THREAD_MUTEX_EXT, XSI_THREADS_EXT:

pthread_mutexattr_gettype(), pthread_mutexattr_settype()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

All APIs in this group behave as defined in POSIX.1-2001.

6.3.31 ELC_REGEXP

(Regular Expressions) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_REGEXP:

regcomp(), regerror(), regexec(), regfree()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.32 ELC_SHELL_FUNC

(Shell and Utilities) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_SHELL_FUNC:

pclose(), popen(), system(), wordexp(), wordfree()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.33 ELC_SIGNALS

(Signal) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_SIGNALS:

abort(), alarm(), kill(), pause(), raise(), sigaction(), sigaddset(), sigdelset(), sigemptyset(), sigfillset(), sigismember(), signal(), sigpending(), sigprocmask(), sigsuspend(), sigwait()

The set of APIs described in SUSv3 Appendix E.1, XSI_SIGNALS:

bsd_signal(), killpg(), sigaltstack(), sighold(), sigignore(), siginterrupt(), sigpause(), sigrelse(), sigset(), ualarm()

The following APIs as defined in LSB1.2:

psignal(), sigandset(), sigblock(), siggetmask(), sigisemptyset(), sigorset(), sigreturn(),

With the exception of the following APIs, which are excluded from this set: None

6.3.34 ELC_SIGNAL_JUMP

(Signal Jump Functions) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_SIGNAL_JUMP:

siglongjmp(), sigsetjmp()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.35 ELC_SINGLE_PROCESS

(Single Process) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_SINGLE_PROCESS:

confstr(), environ, errno, getenv(), setenv(), sysconf(), uname(), unsetenv()

The set of APIs described in SUSv3 Appendix E.1, XSI_SINGLE_PROCESS:

gethostid(), gettimeofday(), putenv()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.36 ELC_STDIO_LOCKING

(Thread-Safe stdio Locking) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_FILE_LOCKING:

flockfile(), ftrylockfile(), funlockfile(), getc_unlocked(), getchar_unlocked(), putc_unlocked(), putchar_unlocked()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.37 ELC_SYMBOLIC_LINKS

(Symbolic Links) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_SYMBOLIC_LINKS:

lstat(), readlink(), symlink()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.38 ELC_SYSTEM_DATABASE

(System Database) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_SYSTEM_DATABASE:

getgrgid(), getgrnam(), getpwnam(), getpwuid()

The set of APIs described in SUSv3 Appendix E.1, XSI_SYSTEM_DATABASE:

endpwent(), getpwent(), setpwent()

The following APIs as defined in LSB1.2:

setmntent(),

With the exception of the following APIs, which are excluded from this set: None

6.3.39 ELC_SYSTEM_DATABASE_R

(Thread-Safe System database) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_SYSTEM_DATABASE_R:

getgrgid_r(), getgrnam_r(), getpwnam_r(), getpwuid_r()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

6.3.40 ELC_SYSTEM_LOGGING

(System Logging) contains

The set of APIs described in SUSv3 Appendix E.1, XSI_SYSTEM_LOGGING:

closelog(), openlog(), setlogmask(), syslog()

The following APIs as defined in LSB1.2:

acct()

With the exception of the following APIs, which are excluded from this set: None

6.3.41 ELC_USER_GROUPS

(User and Group) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_USER_GROUPS:

getegid(), geteuid(), getgid(), getgroups(), getlogin(), getuid(), setegid(), seteuid(), setgid(), setuid()

The set of APIs described in SUSv3 Appendix E.1, XSI_USER_GROUPS:

endgrent(), endutxent(), getgrent(), getutxent(), getutxid(), getutxline(), pututxline(), setgrent(), setregid(), setreuid(), setutxent()

The following APIs as defined in LSB1.2:

initgroups(), getutent(), setgroups(), setutent(),

With the exception of the following APIs, which are excluded from this set: None

6.3.42 ELC_USER_GROUPS_R

(Thread-Safe User and Group) contains

The set of APIs described in POSIX.1-2001 Appendix E.1, POSIX_USER_GROUPS_R:

getlogin_r()

The following APIs as defined in LSB1.2:

getutent_r(),

With the exception of the following APIs, which are excluded from this set: None

6.3.43 ELC_WIDE_CHAR

(Wide Character Library) contains

The set of APIs described in SUSv3 Appendix E.1, XSI_WIDE_CHAR:

wcswidth(), wcwidth()

The following APIs as defined in LSB1.2:

mbsnrtowcs(), wcpcpy(), wcpncpy(), wcscasecmp(), wcsncasecmp(), wcsdup(), wcsnlen(), wcsnrtombs(), wcstoq(), wcstouq(),

With the exception of the following APIs, which are excluded from this set: None

6.3.44 ELC_WIDE_CHAR_DEVICE_IO

(Wide Character Device Input/Output) contains

The set of APIs described in POSIX.1-2001 Appendix E.1,
POSIX_WIDE_CHAR_DEVICE_IO:

fgetwc(), fgetws(), fputwc(), fputws(), fwide(), fwprintf(), fwscanf(), getwc(), getwchar(), putwc(), putwchar(), ungetwc(), vfwprintf(), vfwscanf(), vwprintf(), vwscanf(), wprintf(), wscanf()

The following APIs as defined in LSB1.2: None

With the exception of the following APIs, which are excluded from this set: None

7 Feature Macros and Constants

7.1 Location

A conforming implementation shall make available an <elcstd.h> header, defining the symbolic constants and types described in this section. The actual values of the constants are unspecified except as shown.

7.2 Version Test Macro

The following symbolic constants shall be defined in <elcstd.h>:

_ELCPS_VERSION

Long integer value indicating version of ELCPS to which the implementation conforms. For implementations conforming to this particular version, the value shall be 200212L.

7.3 Constants for Environments and Function/Feature Groups

The following symbolic constants shall be defined in *<elcstd.h>* and shall have a value of -1, 0, or greater, unless otherwise specified below.

If a symbolic constant is defined with the value -1, the option is not supported. Headers, data types, and function interfaces required only for the option need not be supplied. An application that attempts to use anything associated only with the option is considered to be requiring an extension.

If a symbolic constant is defined with a value greater than zero, the option shall always be supported when the application is executed. All headers, data types, and functions shall be present and shall operate as specified.

If a symbolic constant is defined with the value zero, all headers, data types, and functions shall be present. The application can check at runtime to see whether the option is supported by calling *fpathconf()*, *pathconf()*, or *sysconf()* with the indicated name parameter.

Unless explicitly specified otherwise, the behavior of functions associated with an unsupported option is unspecified, and an application that uses such functions without first checking *fpathconf()*, *pathconf()*, or *sysconf()* is considered to be requiring an extension.

_ELCPS_MINIMAL_ENV

The implementation supports the Minimal System Environment. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELCPS_INTERMEDIATE_ENV

The implementation supports the Intermediate System Environment. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELCPS_FULL_ENV

The implementation supports the Full System Environment. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_ASYNCHRONOUS_IO

The implementation supports the Asynchronous I/O interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_C_LANG_JUMP

The implementation supports the ISO C Library Jump Functions interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_C_LANG_MATH

The implementation supports the Math Functions interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_C_LANG_SUPPORT

The implementation supports the General ISO C Library interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_C_LANG_SUPPORT_R

The implementation supports the Thread-Safe General ISO C Library interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_C_LIB_EXT

The implementation supports the General C Library Extension interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_DEVICE_IO

The implementation supports the Device Input and Output interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_DEVICE_SPECIFIC

The implementation supports the General Terminal interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_DEVICE_SPECIFIC_R

The implementation supports the Thread-Safe General Terminal interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_DYNAMIC_LINKING

The implementation supports the Dynamic Linking interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_FD_MGMT

The implementation supports the File Descriptor Management interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_FIFO _FIFO

The implementation supports the FIFO interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_FILE_ATTRIBUTES

The implementation supports the File Attributes interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_STDIO_LOCKING

The implementation supports the Thread-Safe stdio Locking interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_FILE_SYSTEM

The implementation supports the File System interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_FILE_SYSTEM_EXT

The implementation supports the File System Extensions interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_FILE_SYSTEM_R

The implementation supports the Thread-Safe File System interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_IPC

The implementation supports the Interprocess Communication interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_JOB_CONTROL

The implementation supports the Job Control interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_JUMP
The implementation supports the Extended Jump Functions interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_LARGE_FILE
The implementation supports the Large File Support interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_LSB_THREADS
The implementation supports the LSB-Threads interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_LSB_THREADS_EXT
The implementation supports the LSB-Threads Extensions interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_MEM_MGMT
The implementation supports the Memory Management interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_MULTI_ADDR_SPACE
The implementation supports the Multiple Address Space interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_MULTI_PROCESS
The implementation supports the Multiple Processes interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_NETWORKING
The implementation supports the Networking interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_NETWORKING_RPC
The implementation supports the RPC interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_PIPE
The implementation supports the Pipe interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_POSIX_THREADS
The implementation supports the POSIX-Threads interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_POSIX_THREADS_EXT

The implementation supports the POSIX-Threads Extensions interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_REGEXP

The implementation supports the Regular Expressions interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_SC_MIN_ENV

The value returned from sysconf() for _SC_ELCPS_ENVIRONMENT when operating in the Minimal Environment. This value is implementation-defined.

_ELC_SC_INTER_ENV

The value returned from sysconf() for _SC_ELCPS_ENVIRONMENT when operating in the Intermediate Environment. This value is implementation-defined.

_ELC_SC_FULL_ENV

The value returned from sysconf() for _SC_ELCPS_ENVIRONMENT when operating in the Full Environment. This value is implementation-defined.

_ELC_SHELL_FUNC

The implementation supports the Shell and Utilities interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_SIGNALS

The implementation supports the Signals interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_SIGNAL_JUMP

The implementation supports the Signal Jump Functions interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_SINGLE_PROCESS

The implementation supports the Single Process interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_SYMBOLIC_LINKS

The implementation supports the Symbolic Links interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_SYSTEM_DATABASE

The implementation supports the System Database interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_SYSTEM_DATABASE_R

The implementation supports the Threads-safe System Database interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_SYSTEM_LOGGING

The implementation supports the System Logging interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_USER_GROUPS

The implementation supports the User and Group interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_USER_GROUPS_R

The implementation supports the Thread-safe User and Group interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_WIDE_CHAR

The implementation supports the Wide Character Library interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

_ELC_WIDE_CHAR_DEVICE_IO

The implementation supports the Wide Character Device I/O interface group. If this symbol has a value other than -1 or 0, it shall have the value 200212L.

7.4 Dynamic Determination of Environment

The following symbolic constants are defined for *sysconf()*:

_SC_ELCPS_ENVIRONMENT

This constant is used for determination of the environment in which the process is executing.

8 Rationale

This section is for informational purposes only, and is not a part of the normative text of this specification.

The Embedded Linux Consortium Platform Specification (ELCPS) was created with the intent of providing a rationalization of existing formal and de facto standards in the Linux community, for use by embedded systems implementers who are considering (or using) Linux as a development base. As such, it relies heavily on documented standards but modifies and subsets them as necessary for the purposes of this group.

8.1 Use of Existing Standards

The ELCPS relies heavily on the Linux Standards Base, IEEE POSIX, and the Open Group Single UNIX Specifications. Some of the goals of this specification are

- That the specification is compatible with the LSB1.2 specification – that there are no conflicts between the two.
- An implementation conforming to the LSB1.2 can also be called conforming to at least one of the environments described in this specification.

That there is no conflict between this specification and the IEEE POSIX realtime feature sets, as many embedded implementations also use realtime.

8.2 Realtime

The lack of specification concerning IEEE POSIX Realtime Options in this document is intentional. While one may consider the base API specifications in this area "settled" with the approval of IEEE 1003.1-2001 in December 2001, in fact this is still a rapidly-evolving area both in practice and within the POSIX standards community. An additional cause for caution in this area is the total lack of specification or standardization within Linux -- the LSB does not go into detail because it does not follow the POSIX realtime specification. Therefore, we think that there is no established realtime standard for Linux at present.

It is expected that in future versions of the ELCPS, IEEE POSIX Realtime options will be added to the environments or new environments created that require these APIs.

8.3 Threads

The ELCPS has not taken a position concerning threads implementation. The two pieces of the threads implementation are the library and the OS kernel. A commonly used Linux library is the Free Software Foundation GNU C library, which contains a mostly-POSIX-conforming threads API. The Linux kernel, however, is not designed (at the time of ELCPS Version 1.0 publication) to operate threads according to the POSIX model. This means, as the LSB1.2 points out, that

Linux threads are POSIX-conforming with a long list of caveats, a few of which are severe enough to mean that Linux threads are not really usable in a POSIX sense.

However, many markets where embedded Linux would compete, require fully-compliant POSIX threads. There are a few projects underway (such as IBM's Next Generation Pthreads project) that would allow a plugin replacement for the threads package in the GNU library, but these are not available at this time in a manner that provides full POSIX conformance. The ELC solution to this dilemma is to allow an implementer to choose either the default Linux threads package, offer an alternative package, or both. In this way Linux compatibility and marketplace needs can be met.

It is worth noting that this specification assumes that any single application will only use one thread model per that process' lifetime. It also assumes that sets of cooperating applications will need to agree on a single thread model as well. It is not the intent to preclude an implementation offering both models simultaneously, to unrelated processes.

8.4 IPV6

It should be noted that Linux is in constant evolution with new features being added even as the ELCPS is being developed. This standard will also have to evolve to incorporate these changes with future versions. The IPv6 standard is one such example. At the current time, IPv6 is not widely used in embedded systems nor is there a significant infrastructure requiring IPv6 as there is for IPv4. For this reason IPv6 is not *required* in any of the three environments define by the standard. This does not mean that IPv6 cannot be offered by a vendor of ELCPS compliant products. Instead the inclusion of IPv6 is left *optional*.

9 GNU Free Documentation License

This section not a part of the normative text of this specification, but is the licensing text for it.

Version 1.1, March 2000

9.1 Preamble

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

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

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

9.2 Applicability and definitions

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

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

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

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

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

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

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

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

9.3 Verbatim copying

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

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

9.4 Copying in quantity

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

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

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

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

9.5 Modifications

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

- Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

- List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).

- State on the Title page the name of the publisher of the Modified Version, as the publisher.

- Preserve all the copyright notices of the Document.

- Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

- Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

- Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

- Include an unaltered copy of this License.

- Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

- Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

- In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

- Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

- Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.

- Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

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

9.6 Combining documents

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

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

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

9.7 Collections of documents

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

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

9.8 Aggregation with independent works

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

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

9.9 Translation

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

9.10 Termination

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

9.11 Future revisions of this License

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

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

How to use this License for your documents

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

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

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

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