INTERNATIONAL STANDARDIZED PROFILE

ISO/IEC ISP 15287-2

IEEE Std 1003.13

First edition
2000-03-15

Information technology — Standardized Application Environment Profile —

Part 2: **Posix® Realtime Application Support (AEP)**

Technologies de l'information — Profil d'environnement d'application normalisée —

Partie 2: Support d'application en temps réel POSIX® (AEP)

Reference number
ISO/IEC ISP 15287-2:2000(E)
IEEE
Std 1003.13, 1998 edition

Abstract: This standard is part of the POSIX® series of standardized profiles for open systems. It defines environment profiles for portable realtime applications.

Keywords: AEP, application portability, data processing, environment, open systems, operating system, portable application, POSIX profiles, realtime application environments, realtime environment

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Print: ISBN 0-7381-2319-6 SH94843
PDF: ISBN 0-7381-2320-X SS94843

International Standard ISO/IEC ISP 15287-2:2000
IEEE Std 1003.13-1998

Information Technology—
Standardized Application Environment Profile—
POSIX® Realtime Application Support (AEP)

Sponsor

Portable Applications Standards Committee
of the
IEEE Computer Society

Approved 19 March 1998
IEEE-SA Standards Board

Approved 4 January 2000 by the
International Organization for Standardization
and by the
International Electrotechnical Commission

Adopted as an International Standard by the
International Organization for Standardization
and by the
International Electrotechnical Commission

Published by
The Institute of Electrical and Electronics Engineers, Inc.

American National Standard

International Standardized Profile ISO/IEC ISP 15287-2:2000(E)

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. In addition to developing International Standards, ISO/IEC JTC 1 has created a Special Group on Functional Standardization for the elaboration of International Standardized Profiles.

An International Standardized Profile is an internationally agreed, harmonized document which identifies a standard or group of standards, together with options and parameters, necessary to accomplish a function or a set of functions.

Draft International Standardized Profiles are circulated to national bodies for voting. Publication as an International Standardized Profile requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC ISP 15287 may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standardized Profile ISO/IEC ISP 15287-2 was prepared with the collaboration of

- Asia-Oceania Workshop (AOW);
- European Workshop for Open Systems (EWOS);
- Open Systems Environment Implementors' Workshop (OIW).

ISO/IEC ISP 15287 consists of the following parts, under the general title *Information technology — Standardized Application Environment Profile*:

- *Part 1: PSE 10-HIP — Posix Supercomputing Application Environment Profile*
- *Part 2: Posix® Realtime Application Support (AEP)*

Annexes A and B form a normative part of this part of ISO/IEC ISP 15287. Annex C is for information only.

International Organization for Standardization/International Electrotechnical Commission
Case postale 56 • CH-1211 Genève 20 • Switzerland

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE that have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Contents

TABLES

Introduction

The purpose of this standard is to define realtime application environments based on the ISO/IEC 9945 series of standards. It is intended for realtime systems implementors and realtime applications software developers.

This standard is designed to support building systems where not all the interconnected boxes use the same profile, for example, a hierarchical system where the bottom-level device controllers use the "minimal" profile, the next level up follows the larger "control" profile, and so on. There are interfaces called out for the smaller profiles that make no sense in an isolated box; those interfaces are there solely to support the construction of heterogeneous systems, and systems of communicating peers. Such systems are very common in practice.

To summarize, this standard is embedded in a much larger and widely supported set of standards, which yields benefits during code development, as much development and testing is done on the larger and more comfortable systems. It also may be used in the construction of large and heterogenous systems.

Four profiles have been defined to reflect the wide range of system requirements presented by realtime designs. The intent is to provide a meaningful and coherent set of interfaces that will provide software vendors and consumers with a uniform framework for describing and specifying operating system capabilities. This allows an application writer to construct an application that may be easily moved to a different system that supports the same profile. Similarly, it allows a vendor to claim conformance with an established standard, even if that vendor's implementation does not support the full POSIX feature set.

Initially, the focus of this standard is to provide standardized environments supporting the C language. Options are provided for bindings to the Ada programming language as well as for the C language. Bindings for other languages to these services may be developed and this standard will be updated as appropriate.

Within this document, the term “POSIX/RT-AEP” refers to this standard.

Organization of This Part of IEEE Std 1003.13-1998 (POSIX.13)

This standard is divided into eight elements:

(1) General (Section 1)

(2) Normative references (Section 2)

(3) Definitions (Section 3)

(4) Conventions and abbreviations (Section 4)

(5) Conformance (Section 5)

(6) The various realtime profiles (Sections 6 through 9)

(7) ISPICS requirements (C) (Annex A)

(8) ISPICS requirements (Ada) (Annex B)

References are used to direct the reader to other related sections.

Informative annexes are not part of the standard and are provided for information only. They are provided for guidance and to facilitate understanding.

In publishing this standard, the developers simply intend to provide a yardstick against which various realtime application environments can be measured for conformance. It is not the intent of the developers to measure or rate any products, to reward or sanction any vendors of products for conformance or lack of conformance to this standard, or to attempt to enforce this standard by these or any other means. The responsibility for determining the degree of conformance or lack thereof with this standard rests solely with the individual who is evaluating the product claiming to be in conformance with this standard.

Base Documents

The various realtime application environments described herein are based on the ISO/IEC 9945 and IEEE 1003 family of documents as well as ISO 9899, 1539, 8652, and 8859.

Scenario

This standard is based directly on existing small and/or realtime (typically non-UNIX^{TM}) kernel practice as well as the growing body of practice with POSIX conformant kernels having realtime features. The general approach taken in this standard is to specify interfaces, taken from POSIX, sufficient to deliver the functionality typical of current realtime systems, (see Table 1-19 through Table 1-27).

Each profile is specified with full features, to give users clear direction. Vendors may provide means to configure out those parts that are not needed by specific applications. Vendors wishing to expand on the specified profiles are strongly encouraged to take the added interfaces from other POSIX.13 profiles 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.

Audience

The intended audience for this class of profiles is all persons concerned with an industry-wide standard realtime application environment based on the POSIX suite of standards. This includes at least four groups of people:

(1) Persons buying hardware and software systems.

(2) Persons managing companies that are deciding on future corporate computing directions.

(3) Persons implementing realtime operating systems.

(4) Persons developing realtime applications where portability is a primary objective.

Background

The developers of POSIX/RT-AEP represent a cross section of hardware manufacturers, vendors of operating systems and other software development tools, software designers, consultants, academics, authors, applications programmers, and others. In the course of their deliberations, the developers reviewed related U.S. and international standards, both published and in progress.

Conceptually, POSIX/RT-AEP describes a set of application environment profiles needed for the construction and execution of portable realtime application programs.

The developers of this standard have tried to capture the functionality of existing realtime systems in a reasonable number of profiles that specify predominate application environments. It is felt that these profiles, although not optimum, are a best fit to existing classes of applications and systems.

Features of several commercial realtime kernels were considered. These included **pSOS**™[1], **VRTX32**™[2], and **VxWorks**™[3]. Since these products were commercially successful, they must have addressed a significant market segment. In addition, the uniprocessor subset of VITA's **ORKID** specification, NGCR's "***Tiny Real Time***" (TRT), and the **uITRON** specification were examined. These were proposed standard interfaces for small realtime embedded systems.

The following is a list of features that are representative of current realtime systems and highlights the range of system requirements. While some concepts are common to virtually all implementations (e.g., preemptive, priority-based scheduling), some only apply to smaller systems (e.g., a single address space), and some only to more full-featured systems (e.g., network support, self-hosting).

Basic Realtime Multitasking and Synchronization

— Multiple flows of control

— Preemptive priority scheduling of flows of control

— One address space for all flows of control

— Direct control of location of memory areas

— Inter-thread communications mechanism via message passing (queues)

— Binary and counting semaphores, without priority inheritance

— Mutual exclusion, with optional priority inheritance

— Local or global event flags (one thread awaits multiple things)

— Multiple memory areas, with both fixed- and variable-sized blocks allocation policies

— System time in units of clock ticks

— Timeouts on all blocking services in units of clock ticks

[1] **pSOS** *is a registered trademark of Integrated Systems, Inc.*

[2] **VRTX32** *is a registered trademark of Micro Research, Inc.*

[3] **VxWorks** *is a registered trademark of Wind River Systems, Inc.*

— Hardware interrupt control and support for user interrupt handlers

— Signals

— Exception handling

— Minimal synchronous I/O interface: *open*(), *close*(), *read*(), *write*(), *ioctl*()

— Debugger interface

— No memory protection

— Application runs in privileged (supervisor) mode, if applicable

— Direct I/O, rather than via kernel

— System executable size and memory requirements are major constraints

I/O

Realtime systems supporting I/O generally provide the following features:

— Named I/O devices

— Support for serial I/O lines

— Pipes

— Installable user device drivers

— Memory mapped I/O

Local File System

Realtime systems supporting a file system generally provide the following features:

— Named files

— Hierarchical filesystem (directories)

— Contiguous preallocation of disk space

— May provide media compatibility with another filesystem (e.g. **MS-DOS**™[4] or **RT-11**™[5])

— No user IDs or file protection

Historically, filesystems for embedded realtime systems typically have had a one-level name space, contiguous allocation of disk space, and relatively short filenames. They have not supported an arbitrary hierarchy of named directories, non-contiguous allocation of disk space, or long filenames. They may have had numbered directories (e.g. **RSX-11M**™[6]), or only contiguous allocation of disk space (e.g. **RT-11**™).

However, recent commercial offerings have supported multilevel named directories and both contiguous and non-contiguous disk space allocation. In these implementations, the support of these features with potentially non-

[4] **MS-DOS** *is a registered trademark of Microsoft Corporation.*

[5] **RT-11** *is a registered trademark of Digital Equipment Corporation.*

[6] **RSX-11M** *is a registered trademark of Digital Equipment Corporation.*

deterministic performance does not preclude an application from restricting itself to features with deterministic performance. For example, it is still possible to use contiguous files exclusively. Because it is relatively easy to implement both, and need not interfere with deterministic performance, the working group did not make a distinction between realtime and time-sharing filesystems in this AEP.

Traditional implementations of POSIX.1 filesystems employ a disk buffer cache to improve performance by reducing the number of physical media accesses, and by reordering the accesses to take advantage of the characteristics of rotating media. These implementations have not made a distinction between the buffering of data transfers [*read*() and *write*()], and directory operations [*creat*(), *link*(), *unlink*(), *mkdir*(), *rmdir*(), *rename*()]. A result of this is that a system crash at an unexpected moment can leave the filesystem in a corrupted state. This situation is usually corrected at the next system reboot by a filesystem checker and recovery program, such as *fsck*(). The checking and correcting of a corrupted filesystem may take a long and variable amount of time to perform, and may require a human operator to monitor its progress. Either of these characteristics would make a filesystem check unacceptable for some embedded realtime applications. It was suggested therefore, that such applications limit their use of directory operations to *safe* times, and that implementations maintain the filesystem in such a way that a filesystem check during reboot is avoided. This was considered, but rejected on the grounds that not all applications would require the capability, and that it was neither specifiable nor testable.

Network Communication

Realtime systems supporting networking generally provide the following features:

— Compatibility with a protocol stack (e.g. TCP/IP)

— May support applications such as FTP, TELNET, TFTP, rcp

Distributed File System

Realtime systems supporting a distributed (non-local) file system generally provide the following features:

— Remote access to a filesystem

— Performance not realtime

Memory Protection

Realtime systems supporting memory protection (typically requiring a memory management unit) generally provide the following features:

— Memory mapping and protection

— Ability to map to special areas of memory (I/O page, frame buffer)

— Typically do not have demand paging for realtime parts

Multiprocessor Support

Realtime systems supporting multiprocessing generally provide one of the following methods:

— **network**
Non-transparent access to remote objects, RPC

— **distributed**
Transparent access to objects, no load-balancing

— **symmetric**
Presence of a global task scheduling queue (may also have local scheduling queues)

Self-Hosting
Realtime systems supporting the capability for program development, text editing, compilation, etc. generally provide the following features:

— Shell

— Text editor

— Compiler, assembler, linker, debugger

— May have user ID protection

Related Standards Activities

Activities to extend this family of POSIX standards to address additional requirements are in progress and similar efforts can be anticipated in the future.

The following areas are under active consideration at this time, or are expected to become active in the near future:[7]

(1) Language-independent service descriptions of ISO/IEC 9945-1:1996.

(2) C, Ada, and FORTRAN Language bindings to (1).

(3) Shell and Utility facilities.

(4) Verification testing methods.

(5) Additional realtime facilities.

(6) Secure/Trusted System considerations.

(7) Network interface facilities.

(8) System Administration.

(9) Graphical User Interfaces.

(10) Profiles describing application- or user-specific combinations of open systems standards for supercomputing, multiprocessor, and batch extensions; and multiuser systems based on historical models.

(11) An overall guide to POSIX-based or related open systems standards and profiles.

[7] *A Standards Status Report that lists all current IEEE Computer Society standards projects is available from the IEEE Computer Society, 1730 Massachusetts Avenue NW, Washington, DC 20036-1903; Telephone: +1 202 371-0101; FAX: +1 202 728-9614. Working drafts of POSIX standards under development are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331 (http://www.standards.ieee.org/).*

Extensions are approved as ‘‘amendments’’ or ‘‘revisions’’ to a base document, following the IEEE and ISO/IEC Procedures. This standard will be extended to accomodate those extensions.

Approved amendments are published separately until the full document is reprinted and such amendments are incorporated in their proper positions.

If you have interest in participating in the PASC working groups addressing these issues, please send your name, address, and phone number to the:

Secretary, IEEE Standards Board
Institute of Electrical and Electronics Engineers, Inc.
P.O. Box 1331
445 Hoes Lane
Piscataway, NJ 08855-1331

and ask to have this forwarded to the chairperson of the appropriate PASC working group. If you have interest in participating in this work at the international level, contact your ISO/IEC national body.

ISO/IEC ISP 15287-2 was prepared by the Systems Services Working Group— Realtime, sponsored by the Portable Applications Standards Committee of the IEEE Computer Society. When the IEEE Standards Board approved this standard on March 19, 1998, the membership of the System Services Working Group was as follows:

Portable Applications Standards Committee (PASC)

Chair:	Lowell Johnson
Vice Chair:	Joseph M. Gwinn
Functional Chairs:	Jay Ashford
	Andrew Josey
	Curtis Royster
Secretary:	Nick Stoughton

IEEE System Services Working Group — Realtime

Chair:	Joseph M. Gwinn
	Susan Corwin (to 1995)
Secretary:	Karen D. Gordon
	Frank Prindle (1996)
	Lee Schermerhorn (to 1994)
Editor:	Bob Luken
Technical Reviewers:	Andrew E. Wheeler, Jr.
	Joseph M. Gwinn
	Karen D. Gordon
Ballot Coordinators:	Andrew E. Wheeler, Jr.
	James T. Oblinger

Working Group

Ray Alderman
Larry Anderson
Pierre-Jean Arcos
Charles R. Arnold
V. Raj Avula
Theodore P. Baker
Todd Bargorek
Robert Barned
Richard M. Bergman
Nawaf Bitar
Steve Brosky
David Butenhof
Hans Petter Christiansen
Dave Cooper
Susan Corwin
Bill Cox
June R. Curtis
Peter Dibble
Christoph Eck
Maryland R. Edwards
Michael Feustel
Bill Gallmeister
Michael González
Karen D. Gordon
Randy Greene
Rick Greer
Joseph M. Gwinn
Steven A. Haaser
Barbara Haleen
Geoffrey R. Hall
Patrick Hebert
Mary R. Hermann
David Hughes
Duane Hughes
Michael B. Jones
Steven Kleiman
Robert Knighten
C. Douglass Locke
Kent Long
Robert D. Luken
Dave Lunger
Bill Maes
James T. Oblinger
Offer Pazy
Carolyn Petersen
Dave Plauger
Arlan Pool
Franklin C. Prindle
Francois Riche
Robert Rose
Gordon W. Ross
Barry Ruzek
Webb Scales
Lee Schermerhorn
Del Swanson
Barry Traylor
Stephen R. Walli
Andrew E. Wheeler, Jr.
David Wilner
John Zolnowsky

The following persons were members of the 1003.13 Balloting Group that approved the standard for submission to the IEEE Standards Board:

Norman Aaronson
Alejandro Alonso-Muñoz
Pierre-Jean Arcos
Charles R. Arnold
Theodore P. Baker
Robert Barned
Jason Behm
Richard M. Bergman
Andy R. Bihain
Shirley Bockstahler-Brandt
Steve Case
Hu Cheng
Hans Pietter Christiansen
Susan Corwin
Donald Cragun
June R. Curtis
Lee Damico
Christoph Eck
James A. Eiler
Philip H. Enslow
Donna K. Fisher
Michel Gien

John Gilbert
Michael González
Karen D. Gordon
Mars J. Gralia
Randy Greene
Joseph M. Gwinn
Steven A. Haaser
Geoffrey R. Hall
Patrick Hebert
Hans H. Heilborn
Duane Hughes
Hal Jespersen
Michael B. Jones
Joe Kelsey
Judy Kerner
Lawrence J. Kilgallen
Martin J. Kirk
Thomas M. Kurihara
Kevin Lewis
C. Douglass Locke
Kent Long
James P. Lonjers
Lee W. Lucas

Robert D. Luken
Dave Lunger
Marshall McKusick
Craig B. Meyers
Diana Norwood
James T. Oblinger
Dave Plauger
Arlan Pool
Franklin C. Prindle
Paul Rabin
Wendy Rauch
Henry H. Robbins
Steven Schwarm
Del Swanson
Sandra Swearingen
James G. Tanner
Mark-Rene Uchida
Andrew E. Wheeler, Jr.
David Wilner
Oren Yuen
John J. Zenor
John Zolnowsky

When the IEEE-SA Standards Board approved this standard on 19 March 1998, it had the following membership:

Richard J. Holleman, *Chair* **Donald N. Heirman**, *Vice Chair*

Judith Gorman, *Secretary*

Satish K. Aggarwal
Clyde R. Camp
James T. Carlo
Gary R. Engmann
Harold E. Epstein
Jay Forster*
Thomas F. Garrity
Ruben D. Garzon

James H. Gurney
Jim D. Isaak
Lowell G. Johnson
Robert Kennelly
E.G. "Al" Kiener
Joseph L. Koepfinger*
Stephen R. Lambert
Jim Logothetis
Donald C. Loughry

L. Bruce McClung
Louis-François Pau
Ronald C. Petersen
Gerald H. Peterson
John B. Posey
Gary S. Robinson
Hans E. Weinrich
Donald W. Zispe

*Member emeritus

Noelle Humenick
IEEE Standards Project Editor

INTERNATIONAL STANDARD ISO/IEC ISP 15287-2: 2000

IEEE Information Technology — Standardized Application Environment Profile — POSIX® Realtime Application Support (AEP)

Section 1: General

1.1 Overview

This standard establishes a set of Realtime Environment Profiles based on ISO/IEC 9945-1:1990 {3} as amended by IEEE Std 1003.1b-1993 {5}, IEEE Std 1003.5b-1996 {9}, and ISO/IEC 9945-2:1993 {4} (POSIX) work and related standards specifying foundations for realtime applications.

The Application Environment Profiles specified herein are appropriate for the development and execution of realtime applications using the services and utilities provided by standards called out in this document.

1.2 Taxonomy Position

Editor's Note: The following taxonomy is consistent with the 8-25-1993 draft of TR10000-3.

- P — OSE Profiles
  - AEP — Application Environment Profiles
    - PS — System Profiles
      - PSE — Generic Environment Profiles
        - PSE5 — Realtime Environments
          - PSE51 — Minimal Realtime System Profile
          - PSE52 — Realtime Controller System Profile

PSE53 — Dedicated Realtime System Profile

PSE54 — Multi-Purpose Realtime System Profile

1.2.1 Rationale for Positioning (informative)

(This subclause is not a part of ISO/IEC ISP 15287-2: 2000)

This document contains requirements for Application Program Interfaces and Units of Functionality necessary to support four instances of the Generic Realtime Environment class of applications. It specifies the behavior to be observed at the interfaces of the Application Platform on which the class of applications can run. This subset of an OSE profile is complete and coherent within the context of the class of applications supported. As such, it is a System Profile class of Application Environment Profile (AEP).

1.3 Realtime System Profiles

This document describes four realtime profiles and their minimum hardware requirements.

1.3.1 Minimal Realtime System Profile

These systems are typically embedded in systems dedicated to unattended control of one or more special I/O devices. Neither user interaction nor a file system (mass storage) is required. The programming model is that of a single POSIX process (corresponding to the processor's hardware address space) containing one or more threads of control (POSIX.1c threads or Ada tasks). Although there is only one process, a Message Passing interface is provided for communications among threads of control and between PSE5X instantiations. Special devices are operated and controlled either by memory-mapped I/O or by the basic I/O interface, which provides a standard way to access the intrinsically nonstandard I/O hardware and its nonportable control code.

The hardware model for this profile assumes a single processor with its memory, but no memory management unit (MMU) or common I/O devices are required. (If there are in fact multiple processors, typically there are multiple instantiations of the operating system, perhaps communicating via shared memory or a backplane channel, perhaps isolated).

1.3.2 Realtime Controller System Profile

These systems are an extension of the Minimal Realtime System Profile. Support for a file system interface and asynchronous (non-blocking) I/O interfaces has been added.

The hardware model for this profile assumes a single processor and memory (MMU is not required). Mass storage devices are not required; the file system may, for instance, be implemented in memory (RAM disk).

1.3.3 Dedicated Realtime System Profile

These systems are an extension of the Minimal Realtime System Profile. Support for multiple processes has been added. There is a common interface for device drivers and files but no hierarchical file system. Since memory management hardware may be provided, the functionality of memory locking is provided.

The hardware model for this profile assumes one or more processors with or without MMUs in the same system.

1.3.4 Multi-Purpose Realtime System Profile

These systems include all the functionality of the other three profiles. They provide comprehensive functionality and run a mix of differing realtime and non-realtime tasks. This functionality includes all of POSIX.1, POSIX.1b, and POSIX.1c and/or POSIX.5b. Since users may conduct interactive sessions on those systems, all of POSIX.2 and POSIX.2a are also included. Support for additional functionality is provided by options for programming languages. Support for multi-threaded processes is required so that multi-tasking may be done by threads (POSIX.1c threads or Ada tasks), processes, or both.

The hardware model for this profile assumes one or more processors with memory management units, high-speed storage devices, special interfaces, network support, and display devices. The system supports a mix of realtime and non-realtime tasks, some being interactive user tasks.

1.4 Units of Functionality

Some of the profiles specified in this standard do not require support for all the functionality specified in a referenced standard. In this case, if that referenced standard does not contain options for specifying just the required functionality, only those Units of Functionality referenced by the profile may be used by a strictly conforming application. Table 1-1 shows the Units of Functionality defined for POSIX.1; each of these units is a set of functions that represents a separately implementable element of POSIX.1. Table 1-2 defines the functions from the C Standard required to support the POSIX Units of Functionality. Table 1-3 represents the Units of Functionality defined for POSIX.1c. No Units of Functionality are necessary for POSIX.1b. Table 1-5 through Table 1-18 show the Units of Functionality defined for POSIX.5b.

Table 1-1 – POSIX.1 Units of Functionality

Unit of Functionality	Included Functions
POSIX_SINGLE_PROCESS	*sysconf*(), *time*(), *uname*()
POSIX_MULTI_PROCESS	*execl*(), *execle*(), *execlp*(), *execv*(), *execve*(), *execvp*(), *fork*(), *getenv*(), *getpid*(), *getppid*(), *sleep*(), *times*(), *wait*(), *waitpid*(), *_exit*()
POSIX_JOB_CONTROL	*setpgid*(), *tcgetpgrp*(), *tcsetpgrp*()
POSIX_SIGNALS	*alarm*(), *kill*(), *pause*(), *sigaction*(), *sigaddset*(), *sigdelset*(), *sigemptyset*(), *sigfillset*(), *sigismember*(), *sigpending*(), *sigprocmask*(), *sigsuspend*()
POSIX_USER_GROUPS	*geteuid*(), *getegid*(), *getgid*(), *getgroups*(), *getlogin*(), *getpgrp*(), *getuid*(), *setgid*(), *setsid*(), *setuid*()
POSIX_FILE_SYSTEM	*access*(), *chdir*(), *closedir*(), *creat*(), *fpathconf*(), *fstat*(), *getcwd*(), *link*(), *mkdir*(), *opendir*(), *pathconf*(), *readdir*(), *rename*(), *rewinddir*(), *rmdir*(), *stat*(), *unlink*(), *utime*()
POSIX_FILE_ATTRIBUTES	*chmod*(), *chown*(), *umask*()
POSIX_FD_MGMT	*dup*(), *dup2*(), *fcntl*(), *lseek*()
POSIX_DEVICE_IO	*open*(), *close*(), *read*(), *write*()
POSIX_DEVICE_SPECIFIC	*cfgetispeed*(), *cfgetospeed*(), *cfsetispeed*(), *cfsetospeed*(), *ctermid*(), *isatty*(), *tcgetattr*(), *tcsetattr*(), *tcsendbreak*(), *tcdrain*(), *tcflush*(), *tcflow*(), *ttyname*()
POSIX_SYSTEM_DATABASE	*getgrgid*(), *getgrnam*(), *getpwnam*(), *getpwuid*()
POSIX_PIPE	*pipe*()
POSIX_FIFO	*mkfifo*()
POSIX_C_LANG_SUPPORT	See Table 1-4

Table 1-2 – C Language Support for POSIX.1 Units of Functionality

Unit of Functionality	Included Functions
POSIX_DEVICE_IO	*clearerr*(), *getc*(), *getchar*(), *gets*(), *fclose*(), *fdopen*(), *feof*(), *ferror*(), *fflush*(), *fgetc*(), *fgets*(), *fileno*(), *fopen*(), *fprintf*(), *fputc*(), *fputs*(), *fread*(), *freopen*(), *fscanf*(), *fwrite*(), *perror*(), *printf*(), *putc*(), *putchar*(), *puts*(), *sprintf*(), *scanf*(), *sscanf*(), *setbuf*(), *ungetc*()
POSIX_FILE_SYSTEM	*remove*(), *rename*(), *tmpfile*(), *tmpnam*()
POSIX_FD_MGMT	*fseek*(), *ftell*(), *rewind*()
POSIX_MULTI_PROCESS	*assert*(), *exit*(), *setlocale*()
POSIX_SIGNALS	*abort*(), *sigsetjmp*(), *siglongjmp*()

Table 1-3 – POSIX.1c Units of Functionality

Unit of Functionality	Included Functions
POSIX_USER_GROUPS_R	*getlogin_r*()
POSIX_DEVICE_SPECIFIC	*ttyname_r*()
POSIX_FILE_LOCKING	*flockfile*(), *ftrylockfile*(), *funlockfile*(), *getc_unlocked*(), *getchar_unlocked*(), *putc_unlocked*(), *putchar_unlocked*()
POSIX_C_LANG_SUPPORT_R	*strtok_r*(), *asctime_r*(), *ctime_r*(), *gmtime_r*(), *localtime_r*(), *readdir_r*(), *rand_r*()
POSIX_SYSTEM_DATABASE_R	*getgrgid_r*(), *getgrnam_r*(), *getpwuid_r*(), *getpwnam_r*()

Table 1-4 – POSIX_C_LANG_SUPPORT Units of Functionality

Support Group	Included Functions
Character Handling	*isalnum*(), *isalpha*(), *iscntrl*(), *isdigit*(), *isgraph*(), *islower*(), *isprint*(), *ispunct*(), *isspace*(), *isupper*(), *isxdigit*(), *tolower*(), *toupper*()
Mathematics	*acos*(), *asin*(), *atan*(), *atan2*(), *ceil*(), *cos*(), *cosh*(), *exp*(), *fabs*(), *floor*(), *fmod*(), *frexp*(), *idexp*(), *log*(), *log10*(), *modf*(), *pow*(), *sin*(), *sinh*(), *sqrt*(), *tan*(), *tanh*()
Non-Local Jumps	*longjmp*(), *setjmp*()
General Utilities	*abs*(), *atof*(), *atoi*(), *atol*(), *bsearch*(), *calloc*(), *free*(), *malloc*(), *qsort*(), *rand*(), *realloc*(), *srand*()
String Handling	*strcat*(), *strchr*(), *strcmp*(), *strcpy*(), *strcspn*(), *strlen*(), *strncpy*(), *strncat*(), *strncmp*(), *strpbkr*(), *strrchr*(), *strspn*(), *strstr*(), *strtok*()
Date and Time	*asctime*(), *ctime*(), *gmtime*(), *localtime*(), *mktime*(), *strftime*(), *time*(), *tzset*()

Table 1-5 – POSIX.5 Units of Functionality (Single Process)

POSIX_SINGLE_PROCESS	
Package	**Subprograms**
POSIX	All
POSIX_Limits	All
POSIX_Options	All
POSIX_Profiles	All*
POSIX_Configurable_System_Limits	All
POSIX_Calendar	All
POSIX_Process_Environment	Argument_List
	Copy_From_Current_Environment
	Copy_To_Current_Environment
	Copy_Environment
	Clear_Environment
	Set_Environment_Variable
	Delete_Environment_Variable
	Length
	For_Every_Environment_Variable
	For_Every_Current_Environment_Variable
	Environment_Value_Of
	Is_Environment_Variable

NOTE:

* — The package POSIX_Profiles is defined in Annex B of this document.

Table 1-6 – POSIX.5 Units of Functionality (Multi-Process)

POSIX_MULTI_PROCESS	
Package	**Subprograms**
POSIX_Process_Primitives	All
POSIX_Unsafe_Process_Primitives	All
POSIX_Process_Times	All
POSIX_Process_Identification	Get_Process_ID
	Get_Parent_Process_ID

Table 1-7 – POSIX.5 Units of Functionality (Job Control)

POSIX_JOB_CONTROL	
Package	**Subprograms**
POSIX_Process_Identification	Set_Process_Group_ID
	Create_Process_Group
POSIX_Terminal_Functions	Get_Process_Group_ID
	Set_Process_Group_ID
POSIX_Signals	Set_Stopped_Child_Signal
	Stopped_Child_Signal_Enabled

Table 1-8 – POSIX.5 Units of Functionality (Signals)

POSIX_SIGNALS	
Package	**Subprograms**
POSIX_Signals	Add_Signal Add_All_Signals Delete_Signal Delete_All_Signals Is_Member Send_Signal Set_Blocked_Signals Block_Signals Unblock_Signals Blocked_Signals Ignore_Signal Unignore_Signal Is_Ignored Pending_Signals Await_Signal* Await_Signal_Or_Timeout* Interrupt_Task Get_Signal** Set_Signal** Get_Notification Set_Notification Get_Data** Set_Data**

NOTE:

All — indicates all subprograms in a package are required to be supported. Where overloaded versions of a subprogram exist, each instance is required, except as noted. All Image and Value functions must be supported for all packages provided by the implementation.

* — (Return type Signal)

** — (Operation on type Signal_Event)

Table 1-9 – POSIX.5 Units of Functionality (User Groups)

POSIX_USER_GROUPS	
Package	**Subprograms**
POSIX_Process_Identification	Get_Real_User_ID Get_Effective_User_ID Get_Real_Group_ID Get_Effective_Group_ID Set_User_ID Create_Session Set_Group_ID Get_Groups Get_Login_Name Get_Process_Group_ID

Table 1-10 – POSIX.5 Units of Functionality (File System)

POSIX_FILE_SYSTEM	
Package	**Subprograms**
POSIX_Configurable_File_Limits	All
POSIX_File_Status	All
POSIX_Files	For_Every_Directory_Entry
	Create_Directory
	Unlink
	Remove_Directory
	Rename
	Accessibility
	Is_Accessible
	Existence
	Is_File_Present
	Set_File_Times
	Link
	Filename_Of
	Is_File
	Is_Directory
	Is_FIFO
	Is_Character_Special_File
	Is_Block_Special_File
POSIX_Process_Environment	Change_Working_Directory
	Get_Working_Directory
POSIX_IO	Open_Or_Create

Table 1-11 – POSIX.5 Units of Functionality (File Attributes)

POSIX_FILE_ATTRIBUTES	
Package	**Subprograms**
POSIX_Permissions	Set_Allowed_Process_Permissions
	Get_Allowed_Process_Permissions
POSIX_Files	Change_Owner_And_Group
	Change_Permissions

Table 1-12 – POSIX.5 Units of Functionality (FD Management)

POSIX_FD_MGMT	
Package	**Subprograms**
POSIX_File_Locking	All
POSIX_IO	Duplicate
	Duplicate_And_Close
	Get_File_Control
	Set_File_Control
	Get_Close_On_Exec
	Set_Close_On_Exec
	Seek
	File_Size
	File_Position

Table 1-13 – POSIX.5 Units of Functionality (Device IO)

POSIX_DEVICE_IO	
Package	**Subprograms**
POSIX_IO	Open Close Read Write Generic_Read Generic_Write Is_Open

Table 1-14 – POSIX.5 Units of Functionality (Device Specific)

POSIX_DEVICE_SPECIFIC	
Package	**Subprograms**
POSIX_Terminal_Functions:	Get_Terminal_Characteristics Get_Controlling_Terminal_Name Set_Terminal_Characteristics Terminal_Modes_Of Define_Terminal_Modes Bits_Per_Character_Of Define_Bits_Per_Character Special_Control_Character_Of Define_Special_Control_Character Disable_Control_Character Input_Time_Of Define_Input_Time Minimum_Input_Count_Of Define_Minimum_Input_Count Input_Baud_Rate_Of Output_Baud_Rate_Of Define_Input_Baud_Rate Define_Output_Baud_Rate Send_Break Drain Discard_Data Flow
POSIX_IO	Is_A_Terminal Get_Terminal_Name

Table 1-15 – POSIX.5 Units of Functionality (System Database)

POSIX_SYSTEM_DATABASE	
Package	**Subprograms**
POSIX_Group_Database	All
POSIX_User_Database	All

Table 1-16 – POSIX.5 Units of Functionality (Pipes)

POSIX_PIPE	
Package	**Subprograms**
POSIX_IO	Create_Pipe

Table 1-17 – POSIX.5 Units of Functionality (FIFO)

POSIX_FIFO	
Package	**Subprograms**
POSIX_Files	Create_FIFO

Table 1-18 – POSIX.5 Units of Functionality (Ada Language Support)

POSIX_ADA_LANG_SUPPORT	
Package	**Subprograms**
System	Extra Requirements specified in POSIX.5b, Section 2.8
System_Storage_Elements	All
POSIX_Page_Alignment	All
POSIX_Supplement_To_Ada_IO	All
Ada_Task_Identification	All
Ada_Streams	All

1.5 Development Environment

Although POSIX.2/2a is not required for the execution environment of PSE51, PSE52, or PSE53, {POSIX2_SW_DEV} is required in development environments for all four profiles. The options {POSIX2_C_BIND} and {POSIX2_C_DEV} are required for C-Language development environments.

1.6 Summary of Profile Features

The following tables summarize the requirements of the four profiles. Since POSIX.1, POSIX.1c, and/or POSIX.5 do not provide sufficient options to remove features unnecessary for some profiles, Units of Functionality have been developed and are described in Table 1-1 through Table 1-18, and defined by Annex A and Annex B.

Table 1-19 – POSIX.1 and POSIX.5b Units of Functionality Requirements

Unit of Functionality	PSE51	PSE52	PSE53	PSE54
POSIX_SINGLE_PROCESS	X	X	X	X
POSIX_MULTI_PROCESS	-	-	X	X
POSIX_SIGNALS	X	X	X	X
POSIX_USER_GROUPS	-	-	-	X
POSIX_FILE_SYSTEM	-	X	-	X
POSIX_FILE_ATTRIBUTES	-	-	-	X
POSIX_FIFO	-	-	-	X
POSIX_DEVICE_IO	X	X	X	X
POSIX_FD_MGMT	-	X	X	X
POSIX_PIPE	-	-	X	X
POSIX_DEVICE_SPECIFIC	-	-	-	X
POSIX_SYSTEM_DATABASE	-	-	-	X
POSIX_C_LANG_SUPPORT*	X	X	X	X
POSIX_ADA_LANG_SUPPORT**	X	X	X	X

NOTE:

* — Required only for C-Language option.

** — Required only for Ada-Language option.

Table 1-20 – POSIX.1 Option Requirements

Option	PSE51	PSE52	PSE53	PSE54
_POSIX_JOB_CONTROL	-	-	-	X
_POSIX_SAVED_IDS	-	-	-	X
_POSIX_CHOWN_RESTRICTED	-	-	-	X
_POSIX_NO_TRUNC	X	X	X	X
_POSIX_VDISABLE	-	-	-	X
NGROUPS_MAX	-	-	-	>=8

Table 1-21 – POSIX.5b Option Requirements

Option	PSE51	PSE52	PSE53	PSE54
Job_Control_Support	-	-	-	X
Saved_IDs_Support	-	-	-	X
Change_Owner_Restriction	-	-	-	X
Filename_Truncation	-	-	-	-

In all profiles that do not support the POSIX_JOB_CONTROL Unit of Functionality, the subprogram `POSIX_Signals.Set_Stopped_Child_Signal` shall fail silently.

In all profiles that do not support the POSIX_JOB_CONTROL Unit of Functionality, the subprogram `POSIX_Signals.Stopped_Child_Signal_Enabled` shall return `False`.

`POSIX_Limits.Groups_Maxima'First` shall be zero for PSE51, PSE52, and PSE53. For PSE54 it shall be greater than or equal to eight.

`POSIX_Terminal_Functions.Disable_Control_Character` (which corresponds to {_POSIX_VDISABLE}) is not supported in PSE51, PSE52, and PSE53. For PSE54, `POSIX_Terminal_Functions.Disable_Control_Character` shall not raise `POSIX_Error` with an error code of `Operation_Not_Implemented`.

For PSE51 and PSE52, the blocking behavior of all relevant operations defined by POSIX.5b shall be `Tasks` (see POSIX.5b, clause 2.4.1.5).

Table 1-22 – POSIX.1b Option Requirements

Option	PSE51	PSE52	PSE53	PSE54
_POSIX_FSYNC	X	X	X	X
_POSIX_MAPPED_FILES	-	X	-	X
_POSIX_MEMLOCK	X	X	X	X
_POSIX_MEMLOCK_RANGE	X	X	X	X
_POSIX_MEMORY_PROTECTION	-	-	X	X
_POSIX_PRIORITY_SCHEDULING	-	-	X	X
_POSIX_SEMAPHORES	X	X	X	X
_POSIX_SHARED_MEMORY_OBJECTS	X	X	X	X
_POSIX_REALTIME_SIGNALS	X	X	X	X
_POSIX_TIMERS	X	X	X	X
_POSIX_MESSAGE_PASSING	X	X	X	X
_POSIX_SYNCHRONIZED_IO	X	X	X	X
_POSIX_ASYNCHRONOUS_IO	-	X	X	X
_POSIX_PRIORITIZED_IO	-	-	X	X

The correspondence between the options listed in Table 1-22 and the options described in POSIX.5b, clause 2.5, are as follows:

Table 1-23 – POSIX.1b Options vs POSIX.5b Options

POSIX.1b Options	POSIX.5b options
_POSIX_ASYNCHRONOUS_IO	Asynchronous I/O
_POSIX_MAPPED_FILES	Memory Mapped Files
_POSIX_MEMLOCK	Memory Locking
_POSIX_MEMLOCK_RANGE	Memory Range Locking
_POSIX_MEMORY_PROTECTION	Memory Protection
_POSIX_MESSAGE_PASSING	Message Queues
_POSIX_PRIORITIZED_IO	Prioritized I/O
_POSIX_PRIORITY_SCHEDULING	Priority Process Scheduling
_POSIX_REALTIME_SIGNALS	Realtime Signals
_POSIX_SEMAPHORES	Semaphores
_POSIX_SHARED_MEMORY_OBJECTS	Shared Memory Objects
_POSIX_SYNCHRONIZED_IO	Synchronized I/O
_POSIX_TIMERS	Timers
_POSIX_FSYNC	File Synchronization

Table 1-24 – POSIX.1c Option Requirements

Option	PSE51	PSE52	PSE53	PSE54
_POSIX_THREADS	X	X	X	X
_POSIX_THREAD_ATTR_STACKSIZE	X	X	X	X
_POSIX_THREAD_PRIO_INHERIT	X	X	X	X
_POSIX_THREAD_PRIORITY_SCHEDULING	X	X	X	X
_POSIX_THREAD_PRIO_PROTECT	X	X	X	X
_POSIX_THREAD_PROCESS_SHARED	-	-	X	X
_POSIX_THREAD_ATTR_STACKADDR	X	X	X	X

The correspondence between the options listed in Table 1-24 and the options described in POSIX.5b, clause 2.5, are as follows:

Table 1-25 – POSIX.1c Options vs POSIX.5b Options

POSIX.1c Option	POSIX.5b Option
_POSIX_THREAD_PRIO_INHERIT	Mutex Priority Inheritance
_POSIX_THREAD_PRIO_PROTECT	Mutex Priority Ceiling
_POSIX_THREAD_PROCESS_SHARED	Process Shared

NOTE: The other options in Table 1-24 are not visible in the Ada binding. The POSIX.5b analog to the POSIX.1c _POSIX_THREAD_PRIORITY_SCHEDULING option is support for the priority model defined in [Ada RM {1}, clause D.1], the pragmas and package interfaces defined in [Ada RM {1}, clauses D.2-D.5], and the requirements of POSIX.5b, clause 13.2.1.

In addition, the Mutexes Option (specified in POSIX.5b, clause 2.5) must be supported by all profiles.

Table 1-26 – POSIX.1c Unit of Functionality Requirements

Option	PSE51	PSE52	PSE53	PSE54
POSIX_USER_GROUPS_R	-	-	-	X
POSIX_DEVICE_SPECIFIC_R	-	-	-	X
POSIX_FILE_LOCKING	X	X	X	X
POSIX_C_LANG_SUPPORT_R	X	X	X	X
POSIX_SYSTEM_DATABASE_R	-	-	-	X

NOTE: There is no Ada equivalent to Table 1-26, since all POSIX Ada calls are tasking safe.

Table 1-27 – POSIX.2/2a Option Requirements

Option	PSE51**	PSE52**	PSE53**	PSE54
POSIX2_C_BIND*	X	X	X	X
POSIX2_C_DEV*	X	X	X	X
POSIX2_CHAR_TERM	-	-	-	X
POSIX2_FORT_DEV	-	-	-	-
POSIX2_FORT_RUN	-	-	-	X
POSIX2_LOCALEDEF	-	-	-	-
POSIX2_SW_DEV	X	X	X	X
POSIX2_UPE	-	-	-	X

NOTE:

* — Required only for C-Language option.

** — Required only for the development platform.

Section 2: Normative References

2.1 Normative References

The following standards contain provisions which, through references in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this profile of IEEE and ISO are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

{1} ISO/IEC 8652: 1995[1], *Information Technology— Programming languages—Ada [Revision of the first edition (ISO 8652:1987)].*

{2} ISO/IEC 9899: 1990, *Information processing systems—Programming languages—C.*

{3} ISO/IEC 9945-1: 1990 (IEEE Std 1003.1-1990), *Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API) [C Language].*

{4} ISO/IEC 9945-2: 1993 (IEEE Std 1003.2-1992) (includes IEEE Std 1003.2a-1992 User Portability Extension), *Information Technology—Portable Operating System Interface (POSIX)—Part 2: Shell and Utilities.*

{5} IEEE Std 1003.1b-1993, *IEEE Standard for Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API)—Amendment 1: Realtime Extension [C Language].*

{6} IEEE Std 1003.1i-1995, *IEEE Standard for Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API)—Amendment: Technical Corrigenda to Realtime Extension [C Language].*

{7} IEEE Std 1003.5-1992, *IEEE Standard for Information Technology—POSIX Ada Language Interfaces—Part 1: Binding for System Application Program Interface (API).*

[1] ISO/IEC documents can be obtained from the ISO office, 1 rue de Varembé, Case Postale 56, CH-1211, Genève 20, Switzerland/Suisse (http://www.iso.ch/) and from the IEC office, 3 rue de Varembé, Case Postale 131, CH-1211, Genève 20, Switzerland/Suisse (http://www.iec.ch/). ISO/IEC publications are also available in the United States from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA (http://www.ansi.org/).

{8} IEEE Std 1003.1c-1995, *IEEE Standard for Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API)—Amendment 2: Threads Extension [C Language].*

{9} IEEE Std 1003.5b-1996, *IEEE Standard for Information Technology—POSIX Ada Language Interfaces— Part 1: Binding for System Application Program Interface (API)—Amendment 1: Realtime Extensions.*

Section 3: Definitions

3.1 Terms

3.1.1 Terminology

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

3.1.1.1 may: An indication of an optional feature.

With respect to implementations, the word *may* defines optional behavior that any implementation can use and that a strictly conforming application must tolerate. With respect to Strictly Conforming POSIX.13 Applications, the word *may* means that the optional feature shall not be used.

3.1.1.2 shall: An indication of a requirement on the implementation or on Strictly Conforming POSIX.13 Applications, where appropriate.

3.1.1.3 should:

(1) With respect to implementations, an indication of an implementation recommendation, but not a requirement.

(2) With respect to applications, an indication of a recommended programming practice for applications and a requirement for Strictly Conforming POSIX.13 Applications.

3.1.1.4 system documentation: All documentation provided with an implementation, except the conformance document.

Electronically distributed documents for an implementation are considered part of the system documentation.

3.2 Definitions

3.2.1 Terminology

For the purposes of this standard, the following definitions apply:

3.2.1.1 Application Environment Profile (AEP): An OSE profile which specifies a complete and coherent subset of the Open System Environment. [ISO/IEC TR 10000-3:1998 {B3}]

3.2.1.2 Application Platform: A set of resources on which an application will run.

3.2.1.3 Base Standard: An approved IEEE, National, Regional, or International Standard which defines and describes basic functionality and capability. [ISO/IEC TR 10000-1:1998 {B1}]

3.2.1.4 Component Profile: An Application Environment Profile that specifies a unit of functionality in terms of the interfaces that it supports and the interfaces that it uses, and the relationships between these interfaces. [ISO/IEC TR 10000-3:1998 {B3}]

3.2.1.5 Conformance Document: A document provided by an implementor that contains implementation details as described in 5.1.1.2.

3.2.1.6 Development Platform: A system used to prepare an application for execution. Such a system is possibly distinct from the system on which the application will execute.

3.2.1.7 Generic Application Environment Profile: An Application Environment Profile which is not specific to a particular community of use. [ISO/IEC TR 10000-3:1998 {B3}]

3.2.1.8 Generic Interface Profile: An Interface Profile which is not specific to a particular community of use. [ISO/IEC TR 10000-3:1998 {B3}]

3.2.1.9 Industry Specific Application Environment Profile: An Application Environment Profile which deals with specific industry requirements. [ISO/IEC TR 10000-3:1998 {B3}]

3.2.1.10 Industry Specific Interface Profile: An Interface Profile which deals with specific industry requirements. [ISO/IEC TR 10000-3:1998 {B3}]

3.2.1.11 Interface Profile (IP): An OSE Profile defining one interface of the Open System Environment. [ISO/IEC TR 10000-3:1998 {B3}]

3.2.1.12 International Standardized Profile (ISP): An internationally agreed-to, harmonized document which identifies a standard or group of standards, together with options and parameters, necessary to accomplish a function or set of functions. [ISO/IEC TR 10000-1:1998 {B1}]

3.2.1.13 Open System Environment (OSE): The comprehensive set of interfaces, services, and supporting formats for interoperability and/or for portability of applications, data or people, as specified by information technology standards and profiles. [ISO/IEC TR 10000-3:1998 {B3}]

3.2.1.14 Profile (for ISO standardization): A set of one or more base standards (and where applicable) chosen classes, subsets, options, and parameters of those base standards to accomplish a function. [ISO/IEC TR 10000-1:1998 {B1}]

3.2.1.15 Realtime Environment Profile: A profile designed to support applications requiring bounded response.

3.2.1.16 System Profile: An Application Environment Profile that specifies a set of functions necessary to support a class of applications. It specifies the behavior to be observed at the interfaces of the application platform on which the class of applications can run. [ISO/IEC TR 10000-3:1998 {B3}]

NOTE: A system profile is defined in terms of component profiles that specify units of functionality that can be combined to realize the application platform.

3.2.1.17 Unit of Functionality: A separately implementable element of an OSE system. [ISO/IEC TR 10000-3:1998 {B3}]

Section 4: Conventions and Abbreviations

4.1 Conventions

This standard uses the following typographic conventions:

(1) The *italic* font is used for:

- Cross references to defined terms within 5.1, and 3.2.1.
- Symbolic parameters that are generally substituted with real values by the application.
- C-language data types and function names (except in function Synopsis subclauses).
- Global external variable names.

(2) The `constant-width` (Courier) font is used:

- For references to utility names and C language headers.
- For references to Ada identifiers.

(3) Symbolic constants or limits defined in certain headers are represented as:

{_POSIX_AEP_REALTIME_}.

In some cases tabular information is presented ‘‘inline’’; in others it is presented in a separately labeled table. This arrangement was employed purely for ease of typesetting and there is no normative difference between these two cases.

The conventions listed previously are for ease of reading only. Editorial inconsistencies in the use of typography are unintentional and have no normative meaning in this standard.

Notes provided as parts of labeled tables and figures are integral parts of this standard (normative). Footnotes and notes within the body of the text are for information only (informative).

Numerical quantities are presented in international style: comma is used as a decimal sign and units are from the International System (SI).

4.2 Abbreviations

For the purposes of this document the following abbreviations apply:

4.2.1 C Standard: ISO/IEC 9899:1990 {2}, *Information processing systems—Programming languages—C.*

4.2.2 POSIX.1: ISO/IEC 9945-1: 1990 {3}, (IEEE Std 1003.1-1990), *Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API) [C Language].*

4.2.3 POSIX.2 ISO/IEC 9945-2: 1993 {4}, (IEEE Std 1003.2-1992), *Information Technology— Portable Operating System Interface (POSIX)—Part 2: Shell and Utilities.*

4.2.4 POSIX.1b: IEEE Std 1003.1b-1993 {5}, *IEEE Standard for Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API)—Amendment 1: Realtime Extension [C Language],* as amended by IEEE Std 1003.1i-1995 {6}, *IEEE Standard for Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API)—Amendment: Technical Corrigenda to Realtime Extension [C Language].*

4.2.5 POSIX.1c: IEEE Std 1003.1c-1995 {8}, *IEEE Standard for Information Technology—Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API)—Amendment 2: Threads Extension [C Language].*

4.2.6 POSIX.5b: IEEE Std 1003.5-1992 {7}, *IEEE Standard for Information Technology—POSIX Ada Language Interfaces—Part 1: Binding for System Application Program Interface (API) as amended by IEEE Std 1003.5b-1996* {9}, *IEEE Standard for Information Technology—POSIX Ada Language Interfaces—Part 1: Binding for System Application Interface (API)—Amendment 1: Realtime Extensions.*

4.2.7 Ada RM: ISO/IEC 8652: 1995(E) {1}, *Information Technology—Programming Languages—Ada [Revision of the first edition (ISO 8652:1987)].*

4.2.8 AEP: *Application Environment Profile.*

4.2.9 ISP: *International Standardized Profile.*

4.2.10 OSE: *Open System Environment.*

4.2.11 PSE: *Generic Environment Profile.*

4.2.12 PSE51: *The Minimal Realtime System Profile defined herein.*

4.2.13 PSE52: *The Realtime Controller System Profile defined herein.*

4.2.14 PSE53: *The Dedicated Realtime System Profile defined herein.*

4.2.15 PSE54: *The Multi-Purpose Realtime System Profile defined herein.*

Section 5: Conformance

5.1 Conformance

5.1.1 Implementation Conformance

5.1.1.1 Requirements

An implementation may claim conformance to one or more of the profiles defined by this standard. For any given profile a *conforming implementation* shall meet all of the following criteria:

(1) The system shall support all required interfaces referenced in the appropriate standardized profile. These interfaces shall support the functional behavior described in the appropriate base standard, and any additional constraints or options described herein.

(2) The system may provide additional functions or facilities not required by this standard. Nonstandard extensions should be identified as such in the system documentation. Nonstandard extensions, when used, may change the behavior of functions or facilities defined in the appropriate base standard. The conformance document shall define an environment in which an application can be run with predictable behavior specified by the referenced standards. In no case shall such an environment require modification of the appropriate Strictly Conforming POSIX.13 Application.

5.1.1.2 Documentation

An implementation may document conformance to one or more of these profiles in one of two specific manners:

(1) If the implementation is fully conformant to the referenced base standard(s), then that implementation may cite the separate conformance documents that document the base standard conformance. This will primarily apply to implementations that support the PSE53 or PSE54 Profiles.

(2) If the implementation does not fully conform to one or more of the referenced base standards, or if separate base standard conformance documents are not cited, the implementation shall document the specific extent of conformance to each such base standard. This specification shall include:

— A complete list of interfaces from the base standard that are present in the implementation.

— Limit values whose specification is normally required in a conformance document for the base standard (e.g. the limit values found in the <limits.h> and <unistd.h> headers), stating values, the conditions under which those values may change, and the limits of such variations, if any.

— A description of the behavior of the implementation for all implementation-defined features specified by those portions of the base standard that the implementation provides. This requirement shall be met by listing these features and providing either a specific reference to the system documentation or providing full syntax and semantics of these features. The conformance document may specify the behavior of the implementation for those features where the referenced standards state that the implementations may vary or where features are identified as undefined or unspecified.

Regardless of whether separate base standard conformance documents are cited, the conformance document for these profile(s) shall contain a statement that indicates the full name, number, and date of the standard (i.e. the profile standard) that applies. The conformance document may also list international standards that are available for use by a Conforming POSIX.13 Application. Applicable characteristics where documentation is required by one of these standards or by standards of government bodies, may also be included.

5.1.2 Application Conformance

An application claiming conformance to one or more of these profiles shall use only the facilities described in that profile and included referenced standard elements, and shall fall within one of the categories in 5.1.2.1, 5.1.2.2, or 5.1.2.3.

5.1.2.1 Strictly Conforming Application

An application is said to be strictly conforming to a given POSIX.13 profile if the application requires only the facilities required in that profile. Such an application shall accept any behavior described in the profile as *unspecified* or *implementation-defined*, and for symbolic constants, shall accept any value in the range permitted by the profile. Such applications are permitted to adapt to the availability of facilities whose availability is indicated by the constants in 6.1.3, 7.1.3, 8.1.3, and 9.1.3.

5.1.2.2 Conformant Application

5.1.2.2.1 ISO/IEC Conformant Application

An application is said to be ISO/IEC Conformant to a given POSIX.13 profile if the application requires only the facilities required in that profile and approved Conformant Language bindings for any ISO or IEC standard. Such an application shall include a statement of conformance that documents all options and limit dependencies, and all other ISO or IEC standards used.

5.1.2.2.2 <National Body> Conformant POSIX.13 Application

An application is said to be <National Body> Conformant to a given POSIX.13 profile if the application requires only the facilities required in that profile. Such an application shall include a statement of conformance to document all options and limit dependencies, and all other <National Body> standards used.

5.1.2.3 Conformant Application Using Extensions

An application is said to be conformant using extensions if it only uses nonstandard facilities consistent with this standard. Such an application shall fully document its requirements for these extended facilities, in addition to the documentation required of a Conformant Application. A Conformant Application Using Extensions shall be either an ISO/IEC Conformant Application Using Extensions or a <National Body> Conformant Application Using Extensions. (See 5.1.2.2.1 and 5.1.2.2.2.)

Section 6: Minimal Realtime System Profile (PSE51)

6.1 Introduction

This section specifies those standards required for conformance to the Minimal Realtime System Profile option and, where applicable, the state of any options contained in those standards.

When a referenced standard specifies services beyond those required by the Minimal Realtime System Profile, only those services included in the specified Units of Functionality referenced by this profile shall be required. All the applicable definitions in POSIX.1 (as amended by POSIX.1b and POSIX.1c) and/or POSIX.5b still apply. See Table 1-1 and Table 1-5 through Table 1-18.

6.1.1 Identification

For the C-Language implementation, symbolic subtypes shall be used to specify the presence or absence of each option in this profile. Names reserved for use in this profile begin with the string {_POSIX_AEP_REALTIME_}. For the Ada Language implementation a set of Boolean subtypes contained in package `POSIX_Options` (defined in POSIX.5b, section 2.5) shall be used to specify the presence or absence of each option in this profile.

6.1.2 Conformance

Conformance to the Minimal Realtime System Profile option shall be indicated as follows:

For the C-Language implementation the symbol {_POSIX_AEP_REALTIME_MINIMAL} being defined in the header `<unistd.h>`.

For the Ada Language implementation the Boolean subtype `POSIX_Profiles.Realtime_Minimal` having the range `True..True`.

6.1.3 Options

The presence or absence of optional features shall be indicated as follows:

For the C-language implementation, if any of the following symbols are defined in the header `<unistd.h>`:

{_POSIX_AEP_REALTIME_LANG_C89}

{_POSIX_AEP_REALTIME_LANG_Ada95}

For the Ada language implementation, if any of the following Boolean subtypes has the range `True..True`, then the corresponding option is supported :

```
POSIX_Profiles.Realtime_Lang_C89
POSIX_Profiles.Realtime_Lang_Ada95
```

6.2 Operating System Interface Requirements

6.2.1 POSIX.1 Requirements (C Language)

The Minimal Realtime System Profile implementation shall include interfaces as defined in the following portions of POSIX.1 {3}, as amended by POSIX.1b and POSIX.1c:

(1) Process Primitives:

 (a) The POSIX_SIGNALS Unit of Functionality shall be supported.

(2) Process Environment:

 (a) The POSIX_SINGLE_PROCESS Unit of Functionality shall be supported.

(3) Files and Directories:

 (a) The symbol {_POSIX_NO_TRUNC} shall be supported for object names.

(4) Input and Output Primitives:

 (a) The POSIX_DEVICE_IO Unit of Functionality shall be supported.

(5) Language Specific Services:

 (a) The POSIX_C_LANG_SUPPORT Unit of Functionality shall be supported.

6.2.2 POSIX.1b Requirements (C Language)

The Minimal Realtime System Profile implementation shall conform with the following portions of POSIX.1b {5} as amended by POSIX.1c:

(1) File Synchronization:

 (a) The symbol {_POSIX_FSYNC} shall be defined.

(2) Realtime Signals:

 (a) The symbol {_POSIX_REALTIME_SIGNALS} shall be defined.

(3) Synchronized I/O:

 (a) The symbol {_POSIX_SYNCHRONIZED_IO} shall be defined.

(4) Semaphores:

(a) The symbol {_POSIX_SEMAPHORES} shall be defined.

(5) Memory Locking:

(a) The symbol {_POSIX_MEMLOCK} shall be defined.

(b) The symbol {_POSIX_MEMLOCK_RANGE} shall be defined.

(6) Shared Memory:

(a) The symbol {_POSIX_SHARED_MEMORY_OBJECTS} shall be defined.

(7) Clocks and Timers:

(a) The symbol {_POSIX_TIMERS} shall be defined.

(8) Message Passing:

(a) The symbol {_POSIX_MESSAGE_PASSING} shall be defined.

6.2.3 POSIX.1c Requirements (C Language)

The Minimal Realtime System Profile implementation shall conform with the following portions of POSIX.1c {8}:

(1) Threads:

(a) The symbol {_POSIX_THREADS} shall be defined.

(2) Language-Specific Services for the C Programming Language:

(a) The POSIX_FILE_LOCKING Unit of Functionality shall be supported.

(b) The POSIX_C_LANG_SUPPORT_R Unit of Functionality shall be supported.

(3) Synchronization Primitives:

(a) The symbol {_POSIX_THREAD_PRIO_INHERIT} shall be defined.

(b) The symbol {_POSIX_THREAD_PRIO_PROTECT} shall be defined.

(4) Thread Scheduling:

(a) The symbol {_POSIX_THREAD_PRIORITY_SCHEDULING} shall be defined.

(5) Thread Management:

(a) The symbol {_POSIX_THREAD_ATTR_STACKSIZE} shall be defined.

(b) The symbol {_POSIX_THREAD_ATTR_STACKADDR} shall be defined.

6.2.4 POSIX.5b Requirements (Ada Language)

The Minimal Realtime System Profile shall conform with the following portions of POSIX.5b and the Ada RM:

(1) Process Primitives:

(a) The POSIX_SIGNALS Unit of Functionality shall be supported.

(2) Process Environment:

(a) The POSIX_SINGLE_PROCESS Unit of Functionality shall be supported.

(b) The blocking behavior of all reentrant operations defined by POSIX.5b shall be `Tasks`. (See POSIX.5b, clause 2.4.1.5.)

(3) Files and Directories:

(a) The subtype `POSIX_Options.Filename_Truncation` shall have the range `False..False`.

(4) Input and Output Primitives:

(a) The POSIX_DEVICE_IO Unit of Functionality shall be supported.

(5) Language Specific Services:

(a) The POSIX_ADA_LANG_SUPPORT Unit of Functionality shall be supported.

(6) File Synchronization:

(a) The subtype `POSIX_Options.File_Synchronization_Support` shall have the range `True..True`.

(7) Realtime Signals:

(a) The subtype `POSIX_Options.Realtime_Signals_Support` shall have the range `True..True`.

(8) Synchronized I/O:

(a) The subtype `POSIX_Options.Synchronized_IO_Support` shall have the range `True..True`.

(9) Semaphores:

(a) The subtype `POSIX_Options.Semaphores_Support` shall have the range `True..True`.

(10) Memory Locking:

(a) The subtype `POSIX_Options.Memory_Locking_Support` shall have the range `True..True`.

(b) The subtype `POSIX_Options.Memory_Range_Locking_Support` shall have the range `True..True`.

(11) Shared Memory:

(a) The subtype `POSIX_Options.Shared_Memory_Objects_Support` shall have the range `True..True`.

(12) Clocks and Timers:

(a) The subtype `POSIX_Options.Timers_Support` shall have the range `True..True`.

(13) Message Passing:

(a) The subtype `POSIX_Options.Message_Queues_Support` shall have the range `True..True`.

(14) Synchronization Primitives:

(a) The subtype `POSIX_Options.Mutexes_Support` shall have the range `True..True`.

(b) The subtype `POSIX_Options.Mutex_Priority_Inheritance_Support` shall have the range `True..True`.

(c) The subtype `POSIX_Options.Mutex_Priority_Ceiling_Support` shall have the range `True..True`.

(15) Task Scheduling:

(a) The implementation shall support the priority model defined in [Ada RM {1}, clause D.1] and the pragmas and package interfaces defined in [Ada RM {1}, clauses D.2-D.5].

(b) The implementation shall meet the requirements of POSIX.5b, section 13.3.1.

6.3 Rationale for Operating System Requirements (informative)

(This subclause is not a part of ISO/IEC ISP 15287-2: 2000)

6.3.1 POSIX.1 Requirements

After reviewing several commercially available small realtime kernels, it was concluded that the POSIX.1c threads model (with all options enabled, but without a file system) best reflected current industry practice in certain embedded realtime areas. Instead of full file system support, basic device I/O (read, write, open, close, control) is considered sufficient for kernels of this size. Systems of this size frequently do not include process isolation hardware or software; therefore, multiple processes (as opposed to threads) may not be supported.

System options that allow an application to be upwards compatible without modifying application source code have been chosen. For example, although the assumed hardware model implies fixed address space without Memory Management Unit (MMU), the symbol {_POSIX_MEMLOCK} is still defined. This increases portability of the application code to higher level systems that do not necessarily have the same restrictions.

6.3.1.1 Process Primitives

Because this profile uses the POSIX.1c Threads model, most POSIX.1 process primitives do not apply. Signal services are a basic mechanism within POSIX-based systems and are required for error and event handling.

6.3.1.2 Process Environment

Two functions from the POSIX.1 Process Environment group, *sysconf*() and *uname*(), are deemed necessary to allow an application to determine its system environment. This allows a single version of an application to be run on similar but differing platforms, however conforming applications must act as if CHILD_MAX = 0.

6.3.1.3 Files and Directories

The *open*() function is needed to do basic device I/O and also to provide device initialization. Although this requires some form of name resolution, a full pathname space is specifically not required. Directories also are not required. Since a file system is not a part of this realtime profile, the {_POSIX_NO_TRUNC} option is applied to the names of {_POSIX_SHARED_MEMORY_OBJECTS}.

6.3.1.4 Input and Output Primitives

The functions *read*(), *write*(), and *close*() are required to do basic I/O and device cleanup.

6.3.1.5 Device- and Class-Specific Functions

POSIX.1 Device- or Class-Specific functions are not required.

6.3.1.6 Language-Specific Services for the C Programming Language

Support for the C Language is required.

6.3.1.7 System Databases

Implementations are not required to support more than one user and group id since there are not multiple users and groups. No POSIX.1 System Database functions are required.

6.3.2 POSIX.1b Requirements

6.3.2.1 Realtime Signals

These realtime systems typically have several logically concurrent software elements executing. Each such entity must respond to several cyclic and/or acyclic stimuli, often in a time-critical manner. Although purely synchronous models can supply such functionality via the use of additional processes or threads, the current realtime practice for asynchronous notification for events such as timeout, message arrival, and hardware interrupt can generally be expected to offer higher performance and lower latency. Realtime Signals provide the reliable high-performance mechanism to support such notification.

6.3.2.2 Synchronized Input and Output

The Synchronized (unbuffered) I/O interface is typical for basic device I/O and is required for upward portability.

6.3.2.3 Semaphores

Semaphores are required for synchronization between threads to maintain consistency with current industry practice. However, mutexes are preferred in most applications.

6.3.2.4 Process Memory Locking

Process memory locking is inherent in systems following this profile, and is required for upward portability.

6.3.2.5 Shared Memory

Memory Mapped I/O may be implemented using the Shared Memory facility. An implementation shall provide facilities for creating (shared) memory objects that represent ranges of physical memory that contain device control and status registers or buffers. These facilities encourage the development of portable applications.

6.3.2.6 Clocks and Timers

High-resolution timer functions are required in most realtime systems for implementing time management operations such as periodic activations, short duration time-outs, etc. The normal POSIX.1 time management functions *sleep*() and *alarm*() only provide a time resolution of one second, but many realtime systems require higher precision for specifying time.

6.3.2.7 Message Passing

Currently available commercial realtime kernels with similar functionality to the Minimal Realtime System Profile typically include some form of message queueing mechanism for communication between threads. The POSIX.1b message passing option offers an appropriate level of performance to provide this functionality.

6.3.3 POSIX.1c Requirements

The basic assumption in this profile is that the system will consist of a single process, with multiple threads. Therefore, all thread services are required, except for those related to multiple processes.

6.4 Language Requirements

One or more of the development options in 6.4.1 and 6.4.2 shall be implemented.

6.4.1 C Language Development Option

If this option is provided, the implementor shall define a Development Platform and an environment capable of preparing for execution an application conformant with this standard profile. This platform shall include POSIX.2 options {POSIX2_C_BIND}, {POSIX2_C_DEV}, and {POSIX2_SW_DEV}.

6.4.1.1 Option Indicator

The presence of the C Language Development Option shall be indicated by the symbol {_POSIX_AEP_REALTIME_LANG_C89} being defined in the required header `<unistd.h>`. In addition, the presence of the C Language Development Option may be indicated by the subtype `POSIX_Profiles.Realtime_Lang_C89` having the range `True..True`.

6.4.2 Ada Language Development Option

If this option is provided, the implementor shall define a Development Platform and an environment capable of preparing for execution an application conformant with this profile including applicable portions of the following:

— (The Ada RM {1}).

— (POSIX.5b {9}).

— ISO/IEC 9945-1:1993 {4}, (Software Development Option).

6.4.2.1 Option Indicator

The presence of the Ada Language Development Option shall be indicated by the subtype `POSIX_Profiles.Realtime_Lang_Ada95` having the range `True..True`. In addition, the presence of the Ada Language Development Option may be indicated by the symbol {_POSIX_AEP_REALTIME_LANG_Ada95} being defined in the header `<unistd.h>`.

Section 7: Realtime Controller System Profile (PSE52)

7.1 Introduction

This section specifies those standards required for conformance to the Realtime Controller System Profile option and, where applicable, the state of any options contained in those standards.

When a referenced standard specifies services beyond those required by the Realtime Controller System Profile, only those services included in the specified Units of Functionality referenced by this profile shall be required.

7.1.1 Identification

For the C Language implementation, symbolic names shall be used to specify the presence or absence of each option in this profile. Names reserved for use in this profile begin with the string {_POSIX_AEP_REALTIME_}. For the Ada language implementation a set of Boolean subtypes contained in package `POSIX_Options` (defined in POSIX.5b, clause 2.5) shall be used to specify the presence or absence of each option in this profile.

7.1.2 Conformance

Conformance to the Realtime Controller System Profile option shall be indicated as follows:

> For the C language implementation the symbol {_POSIX_AEP_REALTIME_CONTROLLER} being defined in the header `<unistd.h>`.
>
> For the Ada language implementation the Boolean subtype `POSIX_Profiles.Realtime_Controller` having the range `True..True`.

7.1.3 Options

The presence or absence of optional features shall be indicated as follows:

> For the C language implementation, if any of the following symbols are defined in the header `<unistd.h>`, then the corresponding option is supported:
>
> {_POSIX_AEP_REALTIME_LANG_C89}
>
> {_POSIX_AEP_REALTIME_LANG_Ada95}
>
> For the Ada language implementation, if any of the following Boolean subtypes has the range `True..True`, then the corresponding option is

supported :

```
POSIX_Profiles.Realtime_Lang_C89
POSIX_Profiles.Realtime_Lang_Ada95
```

7.2 Operating System Interface Requirements

7.2.1 POSIX.1 Requirements (C language)

The Realtime Controller System Profile implementation shall include interfaces as defined in the following portions of POSIX.1 {3} as amended by POSIX.1b and POSIX.1c:

(1) Process Primitives:

(a) The POSIX_SIGNALS Unit of Functionality shall be supported.

(2) Process Environment:

(a) The POSIX_SINGLE_PROCESS Unit of Functionality shall be supported.

(3) Files and Directories:

(a) The POSIX_FILE_SYSTEM Unit of Functionality shall be supported.

(b) {_POSIX_NO_TRUNC} shall be supported for the native file system.

(4) Input and Output Primitives:

(a) The POSIX_FD_MGMT Unit of Functionality shall be supported.

(b) The POSIX_DEVICE_IO Unit of Functionality shall be supported.

(5) Language Specific Services:

(a) The POSIX_C_LANG_SUPPORT Unit of Functionality shall be supported.

7.2.2 POSIX.1b Requirements (C language)

The Realtime Controller System Profile implementation shall conform with the following portions of POSIX.1b {5} as amended by POSIX.1c:

(1) File Synchronization:

(a) The symbol {_POSIX_FSYNC} shall be defined.

(2) Realtime Signals:

(a) The symbol {_POSIX_REALTIME_SIGNALS} shall be defined.

(3) Synchronized I/O:

(a) The symbol {_POSIX_SYNCHRONIZED_IO} shall be defined.

(4) Asynchronous Input and Output:

(a) The symbol {_POSIX_ASYNCHRONOUS_IO} shall be defined.

(5) Semaphores:

(a) The symbol {_POSIX_SEMAPHORES} shall be defined.

(6) Memory Locking:

(a) The symbol {_POSIX_MEMLOCK} shall be defined.

(b) The symbol {_POSIX_MEMLOCK_RANGE} shall be defined.

(7) Shared Memory:

(a) The symbol {_POSIX_MAPPED_FILES} shall be defined.

(b) The symbol {_POSIX_SHARED_MEMORY_OBJECTS} shall be defined.

(8) Clocks and Timers:

(a) The symbol {_POSIX_TIMERS} shall be defined.

(9) Message Passing:

(a) The symbol {_POSIX_MESSAGE_PASSING} shall be defined.

7.2.3 POSIX.1c Requirements (C language)

The Realtime Controller System Profile implementation shall conform with the following portions of POSIX.1c {8}:

(1) Threads:

(a) The symbol {_POSIX_THREADS} shall be defined.

(2) Language-Specific Services for the C Programming Language:

(a) The POSIX_FILE_LOCKING Unit of Functionality shall be supported.

(b) The POSIX_C_LANG_SUPPORT_R Unit of Functionality shall be supported.

(3) Synchronization Primitives:

(a) The symbol {_POSIX_THREAD_PRIO_INHERIT} shall be defined.

(b) The symbol {_POSIX_THREAD_PRIO_PROTECT} shall be defined.

(4) Thread Scheduling:

(a) The symbol {_POSIX_THREAD_PRIORITY_SCHEDULING} shall be defined.

(5) Thread Management:

(a) The symbol {_POSIX_THREAD_ATTR_STACKSIZE} shall be defined.

(b) The symbol {_POSIX_THREAD_ATTR_STACKADDR} shall be defined.

7.2.4 POSIX.5b Requirements (Ada Language)

The Realtime Controller System Profile shall conform with the following portions of POSIX.5b and the Ada RM:

(1) Process Primitives:

- (a) The POSIX_SIGNALS Unit of Functionality shall be supported.

(2) Process Environment:

- (a) The POSIX_SINGLE_PROCESS Unit of Functionality shall be supported.
- (b) The blocking behavior of all relevant operations defined by POSIX.5b shall be `Tasks` (See POSIX.5b, clause 2.4.1.5).

(3) Files and Directories:

- (a) The POSIX_FILE_SYSTEM Unit of Functionality shall be supported.
- (b) The subtype `POSIX_Options.Filename_Truncation` shall have the range `False..False`.

(4) Input and Output Primitives:

- (a) The POSIX_FD_MGMT Unit of Functionality shall be supported.
- (b) The POSIX_DEVICE_IO Unit of Functionality shall be supported.

(5) Language Specific Services:

- (a) The POSIX_ADA_LANG_SUPPORT Unit of Functionality shall be supported.

(6) File Synchronization:

- (a) The subtype `POSIX_Options.File_Synchronization_Support` shall have the range `True..True`.

(7) Realtime Signals:

- (a) The subtype `POSIX_Options.Realtime_Signals_Support` shall have the range `True..True`.

(8) Synchronized I/O:

- (a) The subtype `POSIX_Options.Synchronized_IO_Support` shall have the range `True..True`.

(9) Asynchronous I/O:

- (a) The subtype `POSIX_Options.Asynchronous_IO_Support` shall have the range `True..True`.

(10) Semaphores:

- (a) The subtype `POSIX_Options.Semaphores_Support` shall have the range `True..True`.

(11) Memory Locking:

- (a) The subtype `POSIX_Options.Memory_Locking_Support` shall have the range `True..True`.

(b) The subtype POSIX_Options.Memory_Range_Locking_Support shall have the range True..True.

(12) Shared Memory:

(a) The subtype POSIX_Options.Memory_Mapped_Files_Support shall have the range True..True.

(b) The subtype POSIX_Options.Shared_Memory_Objects_Support shall have the range True..True.

(13) Clocks and Timers:

(a) The subtype POSIX_Options.Timers_Support shall have the range True..True.

(14) Message Passing:

(a) The subtype POSIX_Options.Message_Queues_Support shall have the range True..True.

(15) Synchronization Primitives:

(a) The subtype POSIX_Options.Mutexes_Support shall have the range True..True.

(b) The subtype POSIX_Options.Mutex_Priority_Inheritance_-Support shall have the range True..True.

(c) The subtype POSIX_Options.Mutex_Priority_Ceiling_Support shall have the range True..True.

(16) Task Scheduling:

(a) The implementation shall support the priority model defined in [Ada RM {1}, clause D.1] and the pragmas and package interfaces defined in [Ada RM {1}, clauses D.2-D.5].

(b) The implementation shall meet the requirements of POSIX.5b, clause 13.2.1.

7.3 Rationale for Operating System Requirements (informative)

(This subclause is not a part of ISO/IEC ISP 15287-2: 2000)

This model introduces more sophisticated system functionality, specifically in the area of I/O. Two general categories of services are added.

The first extension is full support for a file and directory system. These features are used in applications that require an alterable file name space, typically in systems that support secondary storage and require the ability to create, change, and delete named regular files located on a storage device. The included functions allow the creation, deletion, and changing of file attributes of regular files.

The second extension presented in this profile provides the ability to perform asynchronous I/O.

This profile assumes the following hardware model; one or more processors with local memory and one or more serial interfaces. (It is anticipated that the serial interface(s) may be removed in final production systems.) Driver-level I/O to standard and non-standard devices is supported. In addition, a file system device is supported. The hardware is not necessarily expected to provide memory management.

7.3.1 POSIX.1 Requirements

This profile has the same requirements from ISO/IEC 9945-1:1990 {3}, as the Minimal Realtime System Profile, with the addition of file systems and asynchronous I/O.

7.3.1.1 Process Primitives

Because this profile uses the POSIX.1c Threads model, most POSIX.1 process primitives do not apply. The function *main*() is needed to allow application-specific information to be passed from boot code to the single process (and its threads). Signal services are a basic mechanism within POSIX-based systems and are required for error and event handling.

7.3.1.2 Process Environment

Two functions from the POSIX.1 Process Environment group, *sysconf*() and *uname*(), are deemed necessary to allow an application to determine its system environment. This allows a single version of an application to be run on similar but differing platforms.

7.3.1.3 Files and Directories

Since this profile has a file system, all POSIX.1 functions that manage file systems are required, except the following: *umask*(), *chmod*(), *chown*(), and *mkfifo*().

7.3.1.4 Input and Output Primitives

The functions *read*(), *write*(), *close*(), *lseek*(), *dup*(), *dup2*(), and *fcntl*() are required so applications can perform basic I/O device cleanup and file handling.

7.3.1.5 Device- and Class-Specific Functions

POSIX.1 Device- or Class-Specific functions are not required.

7.3.1.6 Language-Specific Services for the C Programming Language

Support for the C Language is required.

7.3.1.7 System Databases

Implementations are not required to support more than one user and group id since there are not multiple users and groups. No POSIX.1 System Database functions are required.

7.3.2 POSIX.1b Requirements

7.3.2.1 Realtime Signals

These realtime systems typically have several logically concurrent software elements executing. Each such entity must respond to several cyclic and/or acyclic stimuli, often in a time-critical manner. Although purely synchronous models can supply such functionality via the use of additional processes or threads, the current realtime practice for asynchronous notification for events such as timeout, message arrival, and hardware interrupt can generally be expected to offer higher performance and lower latency. Realtime Signals provide the reliable high-performance mechanism to support such notification.

7.3.2.2 Synchronized Input and Output

Those realtime systems that use file management systems will frequently require synchronized I/O to provide data integrity and/or relinquish resources to other users. Synchronized I/O as defined in POSIX.1b provides these mechanisms.

7.3.2.3 Asynchronous Input and Output

The application model and existing practice upon which this profile is based requires the ability to do non-blocking I/O on multiple devices.

7.3.2.4 Semaphores

Semaphores are required for synchronization between threads to maintain consistency with current industry practice. However, mutexes are preferred in most applications.

7.3.2.5 Process Memory Locking

Most current implementations of systems with functionality similar to the Realtime Controller Profile do not implement virtual memory, even though a secondary storage device may be present. However, implementations may provide virtual memory and, therefore, process memory locking is required for portability among different implementations, and also for portability to implementations conforming to the the Multi-Purpose Realtime Profile.

7.3.2.6 Shared Memory

Memory Mapped I/O may be implemented using the Shared Memory facility. An implementation must provide facilities for creating (shared) memory objects that represent ranges of physical memory that contain device control and status registers or buffers. These facilities encourage the development of portable applications.

The {_POSIX_MAPPED_FILES} option is included because the implementation has file-system capabilities, and memory-mapped files are a convenient paradigm for reading and writing information in applications following this profile. In memory-mapped files, I/O is not managed by the programmer because data can be manipulated as memory. The implementation of memory-mapped files does not require a significant amount of additional memory or execution overhead to achieve the additional capability.

System vendors are expected to implement the chosen interface in a manner that meets the needs of the applications. In particular, a rotating media-based implementation is not required by the interface definition.

7.3.2.7 Clocks and Timers

High-resolution timer functions are required in most realtime systems for implementing time management operations such as periodic activations, short duration time-outs, etc. The normal POSIX.1 time management functions *sleep*() and *alarm*() only provide a time resolution of one second, but many realtime systems require higher precision for specifying time.

7.3.2.8 Message Passing

Currently available commercial realtime kernels with similar functionality to the Realtime Controller System Profile typically include some form of message queueing mechanism for communication between threads. The POSIX.1b Message Passing offers an appropriate level of performance to provide this functionality.

7.3.3 POSIX.1c Requirements

The basic assumption in this profile is that the system will consist of a single process, with multiple threads. Therefore, all thread services are required, except for those related to multiple processes.

7.4 Language Requirements

One or more of the development options in 7.4.1 and 7.4.2 shall be implemented.

7.4.1 C Language Development Environment

If this option is provided, the implementor shall define a Development Platform and an environment capable of preparing for execution an application conformant with this standard profile. That environment shall include POSIX.2/2a option {POSIX2_SW_DEV}, and for C applications {POSIX2_C_BIND} and {POSIX2_C_DEV}.

7.4.1.1 Option Indicator

The presence of the C Language Development Option shall be indicated by the symbol {_POSIX_AEP_REALTIME_LANG_C89} being defined in the required header `<unistd.h>`. In addition, the presence of the C Language Development Option may be indicated by the subtype `POSIX_Profiles.Realtime_Lang_C89` having the range `True..True`.

7.4.2 Ada Language Development Option

If this option is provided, the implementor shall define a Development Platform and an environment capable of preparing for execution an application conformant with this profile, including applicable portions of the following:

- The Ada RM {1},
- POSIX.5b {9},
- ISO/IEC 9945-2:1993 {4}, (Software Development Option).

7.4.2.1 Option Indicator

The presence of the Ada Language Development Option shall be indicated by the subtype `POSIX_Profiles.Realtime_Lang_Ada95` having the range `True..True`. In addition, the presence of the Ada Language Development Option may be indicated by the symbol {_POSIX_AEP_REALTIME_LANG_Ada95} being defined in the header `<unistd.h>`.

Section 8: Dedicated Realtime System Profile (PSE53)

8.1 Introduction

This section specifies those standards required for conformance to the Dedicated Realtime System Profile option and, where applicable, the state of any options contained in those standards.

When a referenced standard specifies services beyond those required by the Dedicated Realtime System Profile, only those services included in the specified Units of Functionality referenced by this profile shall be required.

8.1.1 Identification

For the C Language implementation, symbolic names shall be used to specify the presence or absence of each option in this profile. Names reserved for use in this profile begin with the string {_POSIX_AEP_REALTIME_}. For the Ada language implementation, a set of Boolean subtypes contained in package `POSIX_Options` (defined in POSIX.5b, clause 2.5) shall be used to specify the presence or absence of each option in this profile.

8.1.2 Conformance

Conformance to the Dedicated Realtime System Profile option shall be indicated as follows:

> For the C language implementation the symbol {_POSIX_AEP_-REALTIME_DEDICATED} being defined in the header `<unistd.h>`.
>
> For the Ada language implementation, the Boolean subtype `POSIX_-Profiles.Realtime_Dedicated` having the range `True..True`.

8.1.3 Options

The presence or absence of optional features shall be indicated as follows:

> For the C language implementation, if any of the following symbols are defined in the header `<unistd.h>`, then the corresponding option is supported:
>
> {_POSIX_AEP_REALTIME_LANG_C89}
>
> {_POSIX_AEP_REALTIME_LANG_Ada95}
>
> For the Ada language implementation, if any of the following Boolean subtypes has the range `True..True`, then the

corresponding option is supported:

```
POSIX_Profiles.Realtime_Lang_C89
POSIX_Profiles.Realtime_Lang_Ada95
```

8.2 Operating System Interface Requirements

8.2.1 POSIX.1 Requirements (C language)

The Dedicated Realtime System Profile implementation shall include interfaces as defined in the following portions of POSIX.1 {3} as amended by POSIX.1b and POSIX.1c:

(1) Process Primitives:

(a) The POSIX_MULTI_PROCESS Unit of Functionality shall be supported.

(b) The POSIX_SIGNALS Unit of Functionality shall be supported.

(2) Process Environment:

(a) The POSIX_SINGLE_PROCESS Unit of Functionality shall be supported.

(3) Files and Directories:

(a) {_POSIX_NO_TRUNC} shall be supported for the native file system.

(4) Input and Output Primitives:

(a) The POSIX_DEVICE_IO Unit of Functionality shall be supported.

(b) The POSIX_FD_MGMT Unit of Functionality shall be supported.

(c) The POSIX_PIPE Unit of Functionality shall be supported.

(5) Language-Specific Services:

(a) The POSIX_C_LANG_SUPPORT Unit of Functionality shall be supported.

8.2.2 POSIX.1b Requirements (C language)

The Dedicated Realtime System Profile implementation shall conform with the following portions of POSIX.1b {5} as amended by POSIX.1c:

(1) File Synchronization:

(a) The symbol {_POSIX_FSYNC} shall be defined.

(2) Realtime Signals:

(a) The symbol {_POSIX_REALTIME_SIGNALS} shall be defined.

(3) Synchronized I/O:

(a) The symbol {_POSIX_SYNCHRONIZED_IO} shall be defined.

(4) Asynchronous Input and Output:

(a) The symbol {_POSIX_ASYNCHRONOUS_IO} shall be defined.

(b) The symbol {_POSIX_PRIORITIZED_IO} shall be defined.

(5) Semaphores:

(a) The symbol {_POSIX_SEMAPHORES} shall be defined.

(6) Memory Locking:

(a) The symbol {_POSIX_MEMLOCK} shall be defined.

(b) The symbol {_POSIX_MEMLOCK_RANGE} shall be defined.

(7) Memory Protection:

(a) The symbol {_POSIX_MEMORY_PROTECTION} shall be defined.

(8) Shared Memory:

(a) The symbol {_POSIX_SHARED_MEMORY_OBJECTS} shall be defined.

(9) Priority Scheduling:

(a) The symbol {_POSIX_PRIORITY_SCHEDULING} shall be defined.

(10) Clocks and Timers:

(a) The symbol {_POSIX_TIMERS} shall be defined.

(11) Message Passing:

(a) The symbol {_POSIX_MESSAGE_PASSING} shall be defined.

8.2.3 POSIX.1c Requirements (C language)

The Dedicated Realtime System Profile implementation shall conform with the following portions of POSIX.1c {8}:

(1) Threads:

(a) The symbol {_POSIX_THREADS} shall be defined.

(2) Language-Specific Services for the C Programming Language:

(a) The POSIX_FILE_LOCKING Unit of Functionality shall be supported.

(b) The POSIX_C_LANG_SUPPORT_R Unit of Functionality shall be supported.

(3) Synchronization Primitives:

(a) The symbol {_POSIX_THREAD_PRIO_INHERIT} shall be defined.

(b) The symbol {_POSIX_THREAD_PRIO_PROTECT} shall be defined.

(c) The symbol {_POSIX_THREAD_PROCESS_SHARED} shall be defined.

(4) Thread Scheduling:

(a) The symbol {_POSIX_THREAD_PRIORITY_SCHEDULING} shall be defined.

(5) Thread Management:

(a) The symbol {_POSIX_THREAD_ATTR_STACKSIZE} shall be defined.

(b) The symbol {_POSIX_THREAD_ATTR_STACKADDR} shall be defined.

8.2.4 POSIX.5b Requirements (Ada Language)

The Dedicated Realtime System Profile shall conform with the following portions of POSIX.5b and the Ada RM:

(1) Process Primitives:

(a) The POSIX_MULTI_PROCESS Unit of Functionality shall be supported, with the provision that the package **POSIX_Unsafe_-Process_Primitives** is not required.

(b) The POSIX_SIGNALS Unit of Functionality shall be supported.

(2) Process Environment:

(a) The POSIX_SINGLE_PROCESS Unit of Functionality shall be supported.

(3) Files and Directories:

(a) The subtype **POSIX_Options.Filename_Truncation** shall have the range **False..False**.

(4) Input and Output Primitives:

(a) The POSIX_FD_MGMT Unit of Functionality shall be supported.

(b) The POSIX_DEVICE_IO Unit of Functionality shall be supported.

(c) The POSIX_PIPE Unit of Functionality shall be supported.

(5) Language Specific Services:

(a) The POSIX_ADA_LANG_SUPPORT Unit of Functionality shall be supported.

(6) File Synchronization:

(a) The subtype **POSIX_Options.File_Synchronization_Support** shall have the range **True..True**.

(7) Realtime Signals:

(a) The subtype **POSIX_Options.Realtime_Signals_Support** shall have the range **True..True**.

(8) Synchronized I/O:

(a) The subtype **POSIX_Options.Synchronized_IO_Support** shall have the range **True..True**.

(9) Asynchronous I/O:

(a) The subtype POSIX_Options.Asynchronous_IO_Support shall have the range True..True.

(b) The subtype POSIX_Options.Prioritized_IO_Support shall have the range True..True.

(10) Semaphores:

(a) The subtype POSIX_Options.Semaphores_Support shall have the range True..True.

(11) Memory Locking:

(a) The subtype POSIX_Options.Memory_Locking_Support shall have the range True..True.

(b) The subtype POSIX_Options.Memory_Range_Locking_Support shall have the range True..True.

(12) Memory Protection:

(a) The subtype POSIX_Options.Memory_Protection_Support shall have the range True..True.

(13) Shared Memory:

(a) The subtype POSIX_Options.Shared_Memory_Objects_Support shall have the range True..True.

(14) Priority Scheduling:

(a) The subtype POSIX_Options.Priority_Process_Scheduling_-Support shall have the range True..True.

(15) Clocks and Timers:

(a) The subtype POSIX_Options.Timers_Support shall have the range True..True.

(16) Message Passing:

(a) The subtype POSIX_Options.Message_Queues_Support shall have the range True..True.

(17) Synchronization Primitives:

(a) The subtype POSIX_Options.Mutexes_Support shall have the range True..True.

(b) The subtype POSIX_Options.Mutex_Priority_Inheritance_-Support shall have the range True..True.

(c) The subtype POSIX_Options.Mutex_Priority_Ceiling_Support shall have the range True..True.

(d) The subtype POSIX_Options.Process_Shared_Support shall have the range True..True.

(18) Task Scheduling:

(a) The implementation shall support the priority model defined in [Ada RM {1}, clause D.1] and the pragmas and package interfaces defined in [Ada RM {1}, clauses D.2-D.5].

(b) The implementation shall meet the requirements of POSIX.5b, clause 13.2.1.

8.3 Rationale for Operating System Requirements (informative)

(This subclause is not a part of ISO/IEC ISP 15287-2: 2000)

This profile is based on existing practice in large embedded systems (single user is assumed). Traditionally, these applications are designed to run with either a home-grown or standard operating system providing process, I/O, time, memory, and event management services. These applications typically do not need full file system support, rather, basic I/O [*read*(), *write*(), *open*(), *close*()] is considered sufficient.

Where convenient, the AEP profile working group has chosen system options that allow an application to be upwardly portable without modifying application source code.

8.3.1 POSIX.1 Requirements

8.3.1.1 Process Primitives

Applications that correspond to the Dedicated Realtime System Environment are usually large embedded systems that require multiple processes for handling multiple, concurrent activities with independent address spaces. The process control functions (which include process creation and execution) are the basic operating system services required to support multiple processes, and are therefore required in these systems.

Signal services are a basic mechanism within POSIX-based systems and are required for error and event handling.

8.3.1.2 Process Environment

Since these systems require multiple processes, but not users or groups, the functions defined by the POSIX_MULTI_PROCESS Unit of Functionality are required.

8.3.1.3 Files and Directories

The *open*() function is needed to do basic device I/O and also to provide device initialization. Although this requires some form of name resolution, a full pathname space is specifically not required. Directories also are not required. Since a file system is not a part of this realtime profile, the {_POSIX_NO_TRUNC} option is applied to the names of {_POSIX_SHARED_MEMORY_OBJECTS}. The *dup*(), and *dup2*() are required for FD manipulation.

8.3.1.4 Input and Output Primitives

The functions *read*(), *write*(), *close*(), *lseek*(), *dup*(), *dup2*(), *fcntl*(), and *pipe*() are required so applications can perform basic I/O and device management.

8.3.1.5 Device- and Class-Specific Functions

No POSIX.1 Device- or Class-Specific functions are required.

8.3.1.6 Language-Specific Services for the C Programming Language

Support for the C Language is required.

8.3.1.7 System Databases

Implementations are not required to support more than one user and group id since there are not multiple users and groups. No POSIX.1 System Database functions are required.

8.3.2 POSIX.1b Requirements

8.3.2.1 Realtime Signals

These realtime systems typically have several logically concurrent software elements executing. Each such entity must respond to several cyclic and/or acyclic stimuli, often in a time critical manner. Although purely synchronous models can supply such functionality via the use of additional processes or threads, the current realtime practice for asynchronous notification for events such as timeout, message arrival and hardware interrupt can generally be expected to offer higher performance and lower latency. Realtime Signals provide the reliable high performance mechanism to support such notification.

8.3.2.2 Synchronized Input and Output

The Synchronized (unbuffered) I/O interface is typical for basic device I/O and is required for upward portability.

8.3.2.3 Asynchronous Input and Output

The application model and existing practice upon which this profile is based requires the ability to do non-blocking I/O on multiple devices.

8.3.2.4 Semaphores

The realtime system synchronization and communications between processes as defined in the Semaphores of POSIX.1b is required. Therefore the option {_POSIX_SEMAPHORES} is required. In this realtime environment, parallel processes are heavily utilized and often must work together to perform a single function. This requires some method of inter-process synchronization which is both fast and simple.

8.3.2.5 Process Memory Locking

Realtime processes must be able to guarantee memory residency to reduce the latency for instruction fetches, data access, I/O operations, etc. The mechanism described in the POSIX.1b Process Memory Locking extension will satisfy this requirement.

8.3.2.6 Shared Memory

The {_POSIX_SHARED_MEMORY_OBJECTS} option provides the capability for more than one execution entity to share memory, without incurring the overhead of the shared memory object on permanent media. Memory Mapped I/O may be implemented using the Shared Memory facility. An implementation must provide facilities for creating a block of physical memory in which the application may place devices and facilities for binding to a user-provided pathname through which a device may subsequently be opened as a Shared Memory special file, and mapped into the process address space for the purposes of performing I/O or other functions from applications programs.

8.3.2.7 Priority Scheduling

This realtime environment requires the ability to do scheduling of concurrent processes with a preemptive priority-based scheduler to ensure that hard deadlines are met.

8.3.2.8 Clocks and Timers

High-resolution timer functions are required in most realtime systems for implementing time management operations such as periodic activations, short duration time-outs, etc. The normal POSIX.1 time management functions *sleep*() and *alarm*() only provide a time resolution of one second, but many realtime systems require higher precision for specifying time.

8.3.2.9 Message Passing

These realtime systems typically include some form of message queuing mechanism for communication among processes or threads. The POSIX.1b message passing offers an appropriate level of performance to provide this functionality.

8.3.3 POSIX.1c requirements

The basic assumption in this profile is that the system will consist of one or more processes with multiple threads. Therefore, all thread services are required.

8.4 Language Requirements

One or more of the development options in 8.4.1 and 8.4.2 shall be implemented.

8.4.1 C Language Development Option

If this option is provided, the implementor shall define a Development Platform and an environment capable of preparing for execution an application conformant with this standard profile. This platform shall include POSIX.2/2a options {POSIX2_C_BIND}, {POSIX2_C_DEV}, and {POSIX2_SW_DEV}.

8.4.1.1 Option Indicator

The presence of the C Language Development Option shall be indicated by the symbol {_POSIX_AEP_REALTIME_LANG_C89} being defined in the required header `<unistd.h>`. In addition, the presence of the C Language Development Option may be indicated by the subtype `POSIX_Profiles.Realtime_Lang_C89` having the range `True..True`.

8.4.2 Ada Language Development Option

If this option is provided, the implementor shall define a Development Platform and an environment capable of preparing for execution an application conformant with this profile, including applicable portions of the following:

- The Ada RM {1},
- POSIX.5b {9},
- ISO/IEC 9945-2:1993 {4}, (Software Development Option).

8.4.2.1 Option Indicator

The presence of the Ada Language Development Option shall be indicated by the subtype `POSIX_Profiles.Realtime_Lang_Ada95` having the range `True..True`. In addition, the presence of the Ada Language Development Option may be indicated by the symbol {_POSIX_AEP_REALTIME_LANG_Ada95} being defined in the header `<unistd.h>`.

Section 9: Multi-Purpose Realtime System Profile (PSE54)

9.1 Introduction

This section specifies those standards required for conformance to the Multi-Purpose Realtime System Profile option and, where applicable, the state of any options contained in those standards.

When a referenced standard specifies services beyond those required by the Multi-Purpose Realtime System Profile, only those services included in the specified Units of Functionality referenced by this profile shall be required.

9.1.1 Identification

For the C Language implementation, symbolic names shall be used to specify the presence or absence of each option in this profile. Names reserved for use in this profile begin with the string {_POSIX_AEP_REALTIME_}. For the Ada language implementation a set of Boolean subtypes contained in package `POSIX_Options` (defined in POSIX.5b, clause 2.5) shall be used to specify the presence or absence of each option in this profile.

9.1.2 Conformance

Conformance to the Multi-Purpose Realtime System Profile option shall be indicated as follows:

For the C language implementation, the symbol {_POSIX_AEP_REALTIME_MULTI} being defined in the header `<unistd.h>`.

For the Ada language implementation the Boolean subtype `POSIX_Profiles.Realtime_Multi` having the range `True..True`.

9.1.3 Options

The presence or absence of optional features shall be indicated as follows:

For the C language implementation, if any of the following symbols are defined in the header `<unistd.h>`, then the corresponding option is supported:

{_POSIX_AEP_REALTIME_LANG_C89}

{_POSIX_AEP_REALTIME_LANG_Ada95}

{_POSIX_AEP_REALTIME_LANG_F77}

{_POSIX_AEP_REALTIME_NETWORK_PPC}

For the Ada language implementation, if any of the following Boolean subtypes has the range `True..True`, then the corresponding option is supported :

```
POSIX_Profiles.Realtime_Lang_C89
POSIX_Profiles.Realtime_Lang_Ada95
POSIX_Profiles.Realtime_Network_PPC
```

9.2 Operating System Interface Requirements

9.2.1 POSIX.1 Requirements (C language)

The Multi-Purpose Realtime System Profile implementation shall include interfaces as defined in the following portions of POSIX.1 {3} as amended by POSIX.1b and POSIX.1c:

(1) Process Primitives:

(a) The POSIX_MULTI_PROCESS Unit of Functionality shall be supported.

(b) The POSIX_SIGNALS Unit of Functionality shall be supported.

(c) The symbol {_POSIX_SAVED_IDS} shall be defined.

(d) The symbol {_POSIX_JOB_CONTROL} shall be defined.

(2) Process Environment:

(a) The POSIX_SINGLE_PROCESS Unit of Functionality shall be supported.

(b) The POSIX_USER_GROUPS Unit of Functionality shall be supported.

(3) Files and Directories:

(a) The POSIX_FILE_SYSTEM Unit of Functionality shall be supported.

(b) The POSIX_FILE_ATTRIBUTES Unit of Functionality shall be supported.

(c) The POSIX_FIFO Unit of Functionality shall be supported.

(d) The symbol {_POSIX_CHOWN_RESTRICTED} shall be defined.

(e) {_POSIX_NO_TRUNC} shall be supported for the native file system.

(4) Input and Output Primitives:

(a) The POSIX_DEVICE_IO Unit of Functionality shall be supported.

(b) The POSIX_FD_MGMT Unit of Functionality shall be supported.

(c) The POSIX_PIPE Unit of Functionality shall be supported.

(5) Device- and Class-Specific Functions:

  (a) The symbol {_POSIX_VDISABLE} shall be defined.

  (b) The POSIX_DEVICE_SPECIFIC Unit of Functionality shall be supported.

(6) Language-Specific Services:

  (a) The POSIX_C_LANG_SUPPORT Unit of Functionality shall be supported.

(7) System Databases:

  (a) The POSIX_SYSTEM_DATABASE Unit of Functionality shall be supported.

9.2.1.1 POSIX.1 Primitive System Data Types

(1) The type *off_t* shall be capable of storing any value contained in type *long*.

9.2.1.2 POSIX.1 Run-Time Increasable Values

(1) The minimum value of {_POSIX_NGROUPS_MAX} shall be at least 8.

9.2.1.3 POSIX.1 Run-Time Invariant Values (Possibly Indeterminate)

(1) The minimum value of {CHILD_MAX} shall be at least 25.

9.2.2 POSIX.1b Requirements (C language)

The Multi-Purpose Realtime System Profile implementation shall conform with the following portions of POSIX.1b {5} as amended by POSIX.1c:

(1) File Synchronization:

  (a) The symbol {_POSIX_FSYNC} shall be defined.

(2) Realtime Signals:

  (a) The symbol {_POSIX_REALTIME_SIGNALS} shall be defined.

(3) Synchronized I/O:

  (a) The symbol {_POSIX_SYNCHRONIZED_IO} shall be defined.

(4) Asynchronous I/O:

  (a) The symbol {_POSIX_ASYNCHRONOUS_IO} shall be defined.

  (b) The symbol {_POSIX_PRIORITIZED_IO} shall be defined.

(5) Semaphores:

  (a) The symbol {_POSIX_SEMAPHORES} shall be defined.

(6) Memory Locking:

(a) The symbol {_POSIX_MEMLOCK} shall be defined.

(b) The symbol {_POSIX_MEMLOCK_RANGE} shall be defined.

(7) Shared Memory:

(a) The symbol {_POSIX_MAPPED_FILES} shall be defined.

(b) The symbol {_POSIX_SHARED_MEMORY_OBJECTS} shall be defined.

(8) Memory Protection:

(a) The symbol {_POSIX_MEMORY_PROTECTION} shall be defined.

(9) Priority Scheduling:

(a) The symbol {_POSIX_PRIORITY_SCHEDULING} shall be defined.

(10) Clocks and Timers:

(a) The symbol {_POSIX_TIMERS} shall be defined.

(11) Message Passing:

(a) The symbol {_POSIX_MESSAGE_PASSING} shall be defined.

9.2.3 POSIX.1c Requirements (C language)

The Multi-Purpose Realtime System Profile implementation shall conform with the following portions of POSIX.1c {8}:

(1) Threads:

(a) The symbol {_POSIX_THREADS} shall be defined.

(2) Process Environment:

(a) The POSIX_USER_GROUPS_R Unit of Functionality shall be supported.

(b) The POSIX_DEVICE_SPECIFIC_R Unit of Functionality shall be supported.

(3) Language-Specific Services for the C Programming Language:

(a) The POSIX_FILE_LOCKING Unit of Functionality shall be supported.

(b) The POSIX_C_LANG_SUPPORT_R Unit of Functionality shall be supported.

(4) System Databases:

(a) The POSIX_SYSTEM_DATABASE_R Unit of Functionality shall be supported.

(5) Synchronization Primitives:

(a) The symbol {_POSIX_THREAD_PRIO_INHERIT} shall be defined.

(b) The symbol {_POSIX_THREAD_PRIO_PROTECT} shall be defined.

(c) The symbol {_POSIX_THREAD_PROCESS_SHARED} shall be defined.

(6) Thread Scheduling:

(a) The symbol {_POSIX_THREAD_PRIORITY_SCHEDULING} shall be defined.

(7) Thread Management:

(a) The symbol {_POSIX_THREAD_ATTR_STACKSIZE} shall be defined.

(b) The symbol {_POSIX_THREAD_ATTR_STACKADDR} shall be defined.

9.2.4 POSIX.5b Requirements (Ada Language)

The Multi-Purpose Realtime System Profile shall conform with the following portions of POSIX.5b and the Ada RM:

(1) Process Primitives:

(a) The POSIX_MULTI_PROCESS Unit of Functionality shall be supported.

(b) The POSIX_SIGNALS Unit of Functionality shall be supported.

(c) The subtype `POSIX_Options.Saved_IDs_Support` shall have the range `True..True`.

(d) The subtype `POSIX_Options.Job_Control_Support` shall have the range `True..True`.

(2) Process Environment:

(a) The POSIX_SINGLE_PROCESS Unit of Functionality shall be supported.

(b) The POSIX_USER_GROUPS Unit of Functionality shall be supported.

(3) Files and Directories:

(a) The POSIX_FILE_SYSTEM Unit of Functionality shall be supported.

(b) The POSIX_FILE_ATTRIBUTES Unit of Functionality shall be supported.

(c) The POSIX_FIFO Unit of Functionality shall be supported.

(d) The subtype `POSIX_Options.Change_Owner_Restriction` shall have the range `True..True`.

(e) The subtype `POSIX_Options.Filename_Truncation` shall have the range `False..False`.

(4) Input and Output Primitives:

(a) The POSIX_FD_MGMT Unit of Functionality shall be supported.

(b) The POSIX_DEVICE_IO Unit of Functionality shall be supported.

(c) The POSIX_PIPE Unit of Functionality shall be supported.

(5) Device- and Class-Specific Services:

(a) The service POSIX_Terminal_Functions.Disable_Control_Character shall not raise POSIX_Error with an error code of Operation_Not_Implemented.

(b) The POSIX_DEVICE_SPECIFIC Unit of Functionality shall be supported.

(6) Language-Specific Services:

(a) The POSIX_ADA_LANG_SUPPORT Unit of Functionality shall be supported.

(7) System Databases:

(a) The POSIX_SYSTEM_DATABASE Unit of Functionality shall be supported.

(8) Minimum Values:

(a) POSIX_Limits.Child_Processes_Maxima'First shall be at least 25.

(b) POSIX_Limits.Groups_Maxima'First shall be at least 8.

(9) File Synchronization:

(a) The subtype POSIX_Options.File_Synchronization_Support shall have the range True..True.

(10) Realtime Signals:

(a) The subtype POSIX_Options.Realtime_Signals_Support shall have the range True..True.

(11) Synchronized I/O:

(a) The subtype POSIX_Options.Synchronized_IO_Support shall have the range True..True.

(12) Asynchronous I/O:

(a) The subtype POSIX_Options.Asynchronous_IO_Support shall have the range True..True.

(b) The subtype POSIX_Options.Prioritized_IO_Support shall have the range True..True.

(13) Semaphores:

(a) The subtype POSIX_Options.Semaphores_Support shall have the range True..True.

(14) Memory Locking:

(a) The subtype POSIX_Options.Memory_Locking_Support shall have the range True..True.

(b) The subtype POSIX_Options.Memory_Range_Locking_Support shall have the range True..True.

(15) Shared Memory:

(a) The subtype `POSIX_Options.Memory_Mapped_Files_Support` shall have the range `True..True`.

(b) The subtype `POSIX_Options.Shared_Memory_Objects_Support` shall have the range `True..True`.

(16) Memory Protection:

(a) The subtype `POSIX_Options.Memory_Protection_Support` shall have the range `True..True`.

(17) Priority Scheduling:

(a) The subtype `POSIX_Options.Priority_Process_Scheduling_-Support` shall have the range `True..True`.

(18) Clocks and Timers:

(a) The subtype `POSIX_Options.Timers_Support` shall have the range `True..True`.

(19) Message Passing:

(a) The subtype `POSIX_Options.Message_Queues_Support` shall have the range `True..True`.

(20) Synchronization Primitives:

(a) The subtype `POSIX_Options.Mutexes_Support` shall have the range `True..True`.

(b) The subtype `POSIX_Options.Mutex_Priority_Inheritance_-Support` shall have the range `True..True`.

(c) The subtype `POSIX_Options.Mutex_Priority_Ceiling_Support` shall have the range `True..True`.

(d) The subtype `POSIX_Options.Process_Shared_Support` shall have the range `True..True`.

(21) Task Scheduling:

(a) The implementation shall support the priority model defined in [Ada RM {1}, clause D.1] and the pragmas and package interfaces defined in [Ada RM {1}, clauses D.2-D.5].

(b) The implementation shall meet the requirements of POSIX.5b, clause 13.2.1.

9.3 Rationale for Operating System Requirements (informative)

(This subclause is not a part of ISO/IEC ISP 15287-2: 2000)

9.3.1 POSIX.1 Requirements

9.3.1.1 Process Primitives

The process control functions (which include process creation and execution) are the basic operating system services required to support multiple processes, and are therefore required by both realtime and non-realtime applications in these realtime systems.

Signal services are a basic mechanism within POSIX-based systems and are required for error and event handling.

9.3.1.2 Process Environment

Since the systems will require multiple processes and multiple users, and because they must support both commercial-off-the-shelf (COTS) and realtime applications, the entire set of ID functions is needed.

9.3.1.3 Files and Directories

All file and directory operations are required to support system applications and their filesystems. Although only a few of the path operation functions are required to support realtime activities, the whole set is required for systems that support COTS applications.

9.3.1.4 Input and Output Primitives

All these functions are required to support I/O on devices, files, and special files.

9.3.1.5 Device-and Class-Specific Functions

The terminal control functions are required for systems to support COTS applications and for the standard terminal devices that may be attached to the computer system. To support non-standard terminal devices, additional functions may be necessary.

9.3.1.6 Language-Specific Service for the C Programming Language

Support for the C language is required.

9.3.1.7 System Databases

The group and user database access functions are required for COTS database applications that may require them.

9.3.2 POSIX.1b Requirements

9.3.2.1 Realtime Signals

These realtime systems typically have several logically concurrent software elements executing. Each such entity must respond to several cyclic and/or acyclic stimuli, often in a time-critical manner. Although purely synchronous models can supply such functionality via the use of additional processes or threads, the current realtime practice for asynchronous notification for events such as timeout, message arrival, and hardware interrupt can generally be expected to offer higher performance and lower latency. Realtime Signals provide the reliable high-performance mechanism to support such notification.

9.3.2.2 Synchronized Input and Output

These realtime systems that use file management systems will frequently require synchronized I/O to provide data integrity and/or relinquish resources to other processes. Synchronized I/O as defined in POSIX.1b provides these mechanisms.

9.3.2.3 Asynchronous Input and Output

The application model and existing practice upon which this profile is based require the ability to do non-blocking I/O on multiple devices.

9.3.2.4 Semaphores

The realtime system synchronization and communications between processes, as defined in the Semaphores of POSIX.1b, are required. Therefore, the option {_POSIX_SEMAPHORES} is required. In this realtime environment, parallel processes are heavily utilized and often must work together to perform a single function. This requires some method of inter-process synchronization that is both fast and simple.

9.3.2.5 Process Memory Locking

Realtime processes must be able to guarantee memory residency to reduce the latency for instruction fetches, data access, I/O operations, etc. The mechanism described in the POSIX.1b Process Memory Locking extension will satisfy this requirement.

9.3.2.6 Shared Memory

The ability to share large volumes of data among many cooperating execution streams is required. The POSIX.1b Shared Memory extension provides this capability. Furthermore, it is sometimes necessary to map special memory locations (refresh buffers, dual-ported memory, I/O devices, etc.) into the address space of one or more execution streams.

The {_POSIX_MAPPED_FILES} option is included because the implementation has file-system capabilities, and memory-mapped files are a convenient paradigm for reading and writing information in applications following this profile. In memory-mapped files, data can be manipulated as memory, and I/O data movement can be significantly reduced. The implementation of memory-mapped files does not require a significant amount of additional memory or execution overhead to achieve the additional capability.

System vendors are expected to implement the chosen interface in a manner that meets the needs of the applications. In particular, a rotating media-based implementation is not required by the interface definition.

9.3.2.7 Priority Scheduling

This realtime environment requires the ability to do scheduling of concurrent processes with a preemptive priority-based scheduler to ensure that hard deadlines are met.

9.3.2.8 Clocks and Timers

High-resolution timer functions are required in most realtime systems for implementing time management operations such as periodic activations, short duration time-outs, etc. The normal POSIX.1 time management functions *sleep*() and *alarm*() only provide a time resolution of one second, but many realtime systems require higher precision for specifying time.

9.3.2.9 Message Passing

These realtime systems typically include some form of message queuing mechanism for communication among processes or threads. The POSIX.1b message passing offers an appropriate level of performance to provide this functionality.

9.3.3 POSIX.1c requirements

The basic assumption in this profile is that the system will consist of one or more processes with multiple threads. Therefore, all thread services are required.

9.4 Shell and Utilities Requirements

Implementations shall conform with the following:

ISO/IEC 9945-2:1993 {4}.

IEEE Std 1003.2-1992 (POSIX 1003.2/2a, Shell and Utilities and User Portability Extension) {4}.

For the POSIX.2/2a options required in this standard, see Tables A-30 and B-21.

9.5 Language Requirements

One or more of the development options in 9.5.1 and 9.5.2 shall be implemented.

9.5.1 C Language Development Option

If this option is provided, the implementor shall define a Development Platform and an environment capable of preparing for execution an application that is conformant with this standard profile.

9.5.1.1 Option Indicator

The presence of the C Language Development Option shall be indicated by the symbol {_POSIX_AEP_REALTIME_LANG_C89} being defined in the required header **<unistd.h>**. In addition, the presence of the C Language Development Option may be indicated by the subtype **POSIX_Profiles.Realtime_Lang_C89** having the range **True..True**.

9.5.2 Ada Language Development Option

If this option is provided, the implementor shall define a Development Platform and an environment capable of preparing for execution an application conformant with this profile, including applicable portions of the following:

- The Ada RM {1}.
- POSIX.5b {9}.
- ISO/IEC 9945-2:1993 {4}, (Software Development Option).

9.5.2.1 Option Indicator

The presence of the Ada Language Development Option shall be indicated by the subtype **POSIX_Profiles.Realtime_Lang_Ada95** having the range **True..True**. In addition, the presence of the Ada Language Development Option may be indicated by the symbol {_POSIX_AEP_REALTIME_LANG_Ada95} being defined in the header **<unistd.h>**.

Annex A
(normative)

ISPICS Requirements List (C Language)

A.1 Options

This document defines four differing profile options that should cover most application environments within the realtime class. These profile options are listed in Table A-1.

Table A-1 – Profile Options

Profile	Symbol	Status
PSE51	{_POSIX_AEP_REALTIME_MINIMAL}	OPT
PSE52	{_POSIX_AEP_REALTIME_CONTROLLER}	OPT
PSE53	{_POSIX_AEP_REALTIME_DEDICATED}	OPT
PSE54	{_POSIX_AEP_REALTIME_MULTI}	OPT

NOTE:

OPT — Optional, one or more profiles may be supported.

A.2 Standards

The following standards are required in whole or in part by one or more of the four included profiles.

Table A-2 – Required Standards

Standard	Profile			
	PSE51	**PSE52**	**PSE53**	**PSE54**
C Standard	PRT	PRT	PRT	MAN
POSIX.1	PRT	PRT	PRT	MAN
POSIX.1b	PRT	PRT	PRT	MAN
POSIX.1c	PRT	PRT	MAN	MAN
POSIX.2/2a	PRT	PRT	PRT	PRT
POSIX.5b	OPT	OPT	OPT	OPT

NOTE:

PRT — Partial, only the subset or options or Units of Functionality called out in A.3.

MAN — Mandatory, complete with all options.

OPT — Optional, may be included in the environment.

A.3 Constraints

Some of the realtime profiles defined in this standard require only specific Units of Functionality of the required standards. The absence of particular elements of these standards introduces constraints on the use of some of the features of particular functions. This clause defines the constraints that an application strictly conforming to one of the profiles shall observe when using each of the functions required by that profile.

A.3.1 POSIX.1

Tables A-3 through A-22 contain the required options, limits, and any other constraints on POSIX.1.

Table A-3 – POSIX.1 Option Requirements

Option	Profile			
	PSE51	PSE52	PSE53	PSE54
{NGROUPS_MAX}	-	-	-	8
{_POSIX_CHOWN_RESTRICTED}	NRQ	NRQ	NRQ	MAN
{_POSIX_JOB_CONTROL}	NRQ	NRQ	NRQ	MAN
{_POSIX_NO_TRUNC}	NAM	PRI	PRI	PRI
{_POSIX_SAVED_IDS}	NRQ	NRQ	NRQ	MAN
{_POSIX_VDISABLE}	NRQ	NRQ	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

NAM — Mandatory to generate an error for object names longer than NAME_MAX.

PRI — The primary file system shall generate an error for pathname components longer than NAME_MAX. The user is responsible for semantics of other file systems that may be mounted.

A.3.1.1 Single Process Function Behavior

The functions in Table A-4 shall behave as described in the referenced clause, except for the *sysconf*() function which has the following constraints:

(1) An application strictly conforming to PSE51, PSE52, PSE53, or PSE54 shall not call the *sysconf*() function with the parameter {_POSIX_VERSION} since a meaningful value cannot be returned.

(2) A conforming application must act as if CHILD_MAX = 0.

Table A-4 – POSIX_SINGLE_PROCESS Functions

Function	Reference in POSIX.1	Status			
		PSE51	PSE52	PSE53	PSE54
sysconf()	4.8.1	MAN	MAN	MAN	MAN
uname()	4.4.1	MAN	MAN	MAN	MAN
time()	4.5.1	MAN	MAN	MAN	MAN

NOTE:

MAN — Mandatory for this profile.

A.3.1.2 Multi-Process Function Behavior

The functions listed in Table A-5 shall behave as described in the referenced clause, except for the *exec*() functions which have the following constraints:

(1) An application strictly conforming to PSE53, which shall use the *path* or *file* argument of *execl*(), *execv*(), *execle*(), *execve*(), *execlp*(), or *execvp*() only to specify the name of the process image to be executed, without any file system semantics implied, because this profile does not require

general file system capabilities.

Table A-5 – POSIX_MULTI_PROCESS Functions

Function	Reference in POSIX.1	Status PSE51	PSE52	PSE53	PSE54
execl ()	3.1.2	NRQ	NRQ	MAN	MAN
execv ()	3.1.2	NRQ	NRQ	MAN	MAN
execle ()	3.1.2	NRQ	NRQ	MAN	MAN
execve ()	3.1.2	NRQ	NRQ	MAN	MAN
execlp ()	3.1.2	NRQ	NRQ	MAN	MAN
execvp ()	3.1.2	NRQ	NRQ	MAN	MAN
_exit ()	3.2.2	NRQ	NRQ	MAN	MAN
fork ()	3.1.1	NRQ	NRQ	MAN	MAN
getenv ()	4.6.1	NRQ	NRQ	MAN	MAN
getpid ()	4.1.1	NRQ	NRQ	MAN	MAN
getppid ()	4.1.1	NRQ	NRQ	MAN	MAN
sleep ()	3.4.3	NRQ	NRQ	MAN	MAN
times ()	4.5.2	NRQ	NRQ	MAN	MAN
wait ()	3.2.1	NRQ	NRQ	MAN	MAN
waitpid ()	3.2.1	NRQ	NRQ	MAN	MAN
assert ()	8.1, 8.2, 8.3	NRQ	NRQ	MAN	MAN
exit ()	8.1, 8.2, 8.3	NRQ	NRQ	MAN	MAN
setlocale ()	8.1, 8.2, 8.3	NRQ	NRQ	MAN	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.1.3 Job Control Function Behavior

The functions listed in Table A-6 shall behave as described in the referenced clause.

Table A-6 – POSIX_JOB_CONTROL Functions

Function*	Reference in POSIX.1	Status PSE51	PSE52	PSE53	PSE54
setpgid ()	4.3.3	NRQ	NRQ	NRQ	MAN
tcgetpgrp ()	7.2.3	NRQ	NRQ	NRQ	MAN
tcsetpgrp ()	7.2.4	NRQ	NRQ	NRQ	MAN
*	7.1.1.4	NRQ	NRQ	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

* — Further functionality is also defined here.

A.3.1.4 Signals Function Behavior

The functions listed in Table A-7 shall behave as described in the referenced clause, except for the following constraints:

(1) An application strictly conforming to PSE51 or PSE52 shall be considered erroneous if any signal results in abnormal termination of the process because these profiles do not support multiple processes.

(2) An application strictly conforming to PSE51, PSE52, or PSE53 shall not call the *kill*() function with a negative argument because these profiles do not require process group functionality.

Table A-7 – POSIX_SIGNALS Functions

Function	Reference in POSIX.1	Status			
		PSE51	PSE52	PSE53	PSE54
alarm()	3.4.1	MAN	MAN	MAN	MAN
kill()	3.3.2	MAN	MAN	MAN	MAN
pause()	3.4.2	MAN	MAN	MAN	MAN
sigaction()	3.3.4	MAN	MAN	MAN	MAN
sigaddset()	3.3.3	MAN	MAN	MAN	MAN
sigdelset()	3.3.3	MAN	MAN	MAN	MAN
sigemptyset()	3.3.3	MAN	MAN	MAN	MAN
sigfillset()	3.2.3	MAN	MAN	MAN	MAN
sigismember()	3.3.3	MAN	MAN	MAN	MAN
sigpending()	3.3.6	MAN	MAN	MAN	MAN
sigprocmask()	3.3.5	MAN	MAN	MAN	MAN
sigsupend()	3.3.7	MAN	MAN	MAN	MAN
abort()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
siglongjmp()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
sigsetjmp()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN

NOTE:

MAN — Mandatory for this profile.

A.3.1.5 User Group Function Behavior

The functions listed in Table A-8 shall behave as described in the referenced clause.

Table A-8 – POSIX_USER_GROUPS Functions

Function	Reference in POSIX.1	Status PSE51	PSE52	PSE53	PSE54
getegid()	4.2.1	NRQ	NRQ	NRQ	MAN
geteuid()	4.2.1	NRQ	NRQ	NRQ	MAN
getgid()	4.2.1	NRQ	NRQ	NRQ	MAN
getgroups()	4.2.3	NRQ	NRQ	NRQ	MAN
getlogin()	4.2.4	NRQ	NRQ	NRQ	MAN
getpgrp()	4.3.1	NRQ	NRQ	NRQ	MAN
getuid()	4.2.1	NRQ	NRQ	NRQ	MAN
setuid()	4.2.2	NRQ	NRQ	NRQ	MAN
setsid()	4.3.2	NRQ	NRQ	NRQ	MAN
setgid()	4.2.2	NRQ	NRQ	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.1.6 File System Function Behavior

The functions listed in Table A-9 shall behave as described in the referenced clause.

Table A-9 – POSIX_FILE_SYSTEM Functions

Function	Reference in POSIX.1	Status PSE51	PSE52	PSE53	PSE54
access()	5.6.3	NRQ	MAN	NRQ	MAN
chdir()	5.2.1	NRQ	MAN	NRQ	MAN
closedir()	5.1.2	NRQ	MAN	NRQ	MAN
creat()	5.3.2	NRQ	MAN	NRQ	MAN
fpathconf()	5.7.1	NRQ	MAN	NRQ	MAN
fstat()	5.6.2	NRQ	MAN	NRQ	MAN
getcwd()	5.2.2	NRQ	MAN	NRQ	MAN
link()	5.3.4	NRQ	MAN	NRQ	MAN
mkdir()	5.4.1	NRQ	MAN	NRQ	MAN
opendir()	5.1.2	NRQ	MAN	NRQ	MAN
pathconf()	5.7.1	NRQ	MAN	NRQ	MAN
readdir()	5.1.2	NRQ	MAN	NRQ	MAN
rename()	5.5.3	NRQ	MAN	NRQ	MAN
rewinddir()	5.1.2	NRQ	MAN	NRQ	MAN
rmdir()	5.5.2	NRQ	MAN	NRQ	MAN
stat()	5.6.2	NRQ	MAN	NRQ	MAN
unlink()	5.5.1	NRQ	MAN	NRQ	MAN
utime()	5.6.6	NRQ	MAN	NRQ	MAN
remove()	8.1, 8.2, 8.3	NRQ	MAN	NRQ	MAN
rename()	8.1, 8.2, 8.3	NRQ	MAN	NRQ	MAN
tmpfile()	8.1, 8.2, 8.3	NRQ	MAN	NRQ	MAN
tmpnam()	8.1, 8.2, 8.3	NRQ	MAN	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.1.7 File Attributes Function Behavior

The functions listed in Table A-10 shall behave as described in the referenced clause, except for the following constraint:

(1) An application strictly conforming to PSE51, PSE52, or PSE53 shall be guaranteed that the file mode creation mask for any object created by any process is `S-IRWXU`; that is, the object shall be fully accessible to the creator.

Table A-10 – POSIX_FILE_ATTRIBUTES Functions

Function	Reference in POSIX.1	Status			
		PSE51	PSE52	PSE53	PSE54
chmod()	5.6.4	NRQ	NRQ	NRQ	MAN
chown()	5.6.5	NRQ	NRQ	NRQ	MAN
umask()	5.3.3	NRQ	NRQ	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.1.8 File and Directory Management Function Behavior

The functions listed in Table A-11 shall behave as described in the referenced clause.

Table A-11 – POSIX_FD_MGMT Functions

Function	Reference in POSIX.1	Status			
		PSE51	PSE52	PSE53	PSE54
dup()	6.2.1	NRQ	MAN	MAN	MAN
dup2()	6.2.1	NRQ	MAN	MAN	MAN
fcntl()	6.5.2	NRQ	MAN	MAN	MAN
lseek()	6.5.3	NRQ	MAN	MAN	MAN
fseek()	8.1, 8.2, 8.3	NRQ	MAN	MAN	MAN
ftell()	8.1, 8.2, 8.3	NRQ	MAN	MAN	MAN
rewind()	8.1, 8.2, 8.3	NRQ	MAN	MAN	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.1.9 Device I/O Function Behavior

The functions listed in Table A-12 shall behave as described in the referenced clause, except for the *fopen*() function which has the following constraints:

(1) An application strictly conforming to PSE51 or PSE53 shall not use the *fopen*() or *freopen*() functions to create new files, since these profiles do not require general file system capabilities.

(2) An application strictly conforming to PSE51 or PSE53 shall use the *path* or *file* argument for *open*() only to specify the name of the object without any file system semantics implied, since these profiles do not require general file system semantics.

(3) An application strictly conforming to PSE51 or PSE53 shall not require that *read*() or *write*() update an access time for the device read or

written, because these profiles require no interfaces that could query such an access time.

Table A-12 – POSIX_DEVICE_IO Functions

Function	Reference in POSIX.1	Status PSE51	PSE52	PSE53	PSE54
close()	6.3.1	MAN	MAN	MAN	MAN
open()	5.3.1	MAN	MAN	MAN	MAN
read()	6.4.1	MAN	MAN	MAN	MAN
write()	6.4.2	MAN	MAN	MAN	MAN
clearerr()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fclose()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fdopen()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
feof()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
ferror()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fflush()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fgetc()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fileno()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fgets()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fopen()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fprintf()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fputc()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fputs()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fread()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
freopen()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fscanf()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
fwrite()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
getc()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
getchar()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
gets()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
perror()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
printf()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
putc()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
putchar()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
puts()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
scanf()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
setbuf()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
sprintf()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
sscanf()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN
ungetc()	8.1, 8.2, 8.3	MAN	MAN	MAN	MAN

NOTE:

MAN — Mandatory for this profile.

A.3.1.10 Device-Specific Function Behavior

The functions listed in Table A-13 shall behave as described in the referenced clause.

Table A-13 – POSIX_DEVICE_SPECIFIC Functions

Function	Reference in POSIX.1	Status			
		PSE51	PSE52	PSE53	PSE54
cfgetispeed ()	7.1.3	NRQ	NRQ	NRQ	MAN
cfgetospeed ()	7.1.3	NRQ	NRQ	NRQ	MAN
cfsetispeed ()	7.1.3	NRQ	NRQ	NRQ	MAN
cfsetospeed ()	7.1.3	NRQ	NRQ	NRQ	MAN
ctermid ()	4.7.1	NRQ	NRQ	NRQ	MAN
isatty ()	4.7.2	NRQ	NRQ	NRQ	MAN
tcdrain ()	7.2.2	NRQ	NRQ	NRQ	MAN
tcflush ()	7.2.2	NRQ	NRQ	NRQ	MAN
tcflow ()	7.2.2	NRQ	NRQ	NRQ	MAN
tcgetattr ()	7.2.1	NRQ	NRQ	NRQ	MAN
tcsendbreak ()	7.2.2	NRQ	NRQ	NRQ	MAN
tcsetattr ()	7.2.1	NRQ	NRQ	NRQ	MAN
ttyname ()	4.7.2	NRQ	NRQ	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.1.11 System Database Function Behavior

The functions listed in Table A-14 shall behave as described in the referenced clause.

Table A-14 – POSIX_SYSTEM_DATABASE Functions

Function	Reference in POSIX.1	Status			
		PSE51	PSE52	PSE53	PSE54
getgrgid ()	9.2.1	NRQ	NRQ	NRQ	MAN
getgrnam ()	9.2.1	NRQ	NRQ	NRQ	MAN
getpwnam ()	9.2.2	NRQ	NRQ	NRQ	MAN
getpwuid ()	9.2.2	NRQ	NRQ	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.1.12 Pipe Function Behavior

The function listed in Table A-15 shall behave as described in the referenced clause.

Table A-15 – POSIX_PIPE Function

Function	Reference in POSIX.1	Status			
		PSE51	PSE52	PSE53	PSE54
pipe()	6.1.1	NRQ	NRQ	MAN	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.1.13 FIFO Function Behavior

The function listed in Table A-16 shall behave as described in the referenced clause.

Table A-16 – POSIX_FIFO Function

Function	Reference in POSIX.1	Status			
		PSE51	PSE52	PSE53	PSE54
mkfifo()	5.4.2	NRQ	NRQ	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.1.14 C Language-Specific Services Behavior

The functions listed in Table A-17, Table A-18, Table A-19, Table A-20, Table A-21, and Table A-22 shall behave as described in the referenced clause.

Table A-17 – POSIX_C_LANG_SUPPORT Character Handling Functions

Function	Reference in the C Standard	Status			
		PSE51	PSE52	PSE53	PSE54
isalnum()	4.3.1.1	MAN	MAN	MAN	MAN
isalpha()	4.3.1.2	MAN	MAN	MAN	MAN
iscntrl()	4.3.1.3	MAN	MAN	MAN	MAN
isdigit()	4.3.1.4	MAN	MAN	MAN	MAN
isgraph()	4.3.1.5	MAN	MAN	MAN	MAN
islower()	4.3.1.6	MAN	MAN	MAN	MAN
isprint()	4.3.1.7	MAN	MAN	MAN	MAN
ispunct()	4.3.1.8	MAN	MAN	MAN	MAN
isspace()	4.3.1.9	MAN	MAN	MAN	MAN
isupper()	4.3.1.10	MAN	MAN	MAN	MAN
isxdigit()	4.3.1.11	MAN	MAN	MAN	MAN
tolower()	4.3.2.1	MAN	MAN	MAN	MAN
toupper()	4.3.2.2	MAN	MAN	MAN	MAN

NOTE:

MAN — Mandatory for this profile.

Table A-18 – POSIX_C_LANG_SUPPORT Mathematical Functions

Function	Reference in the C Standard	Status			
		PSE51	PSE52	PSE53	PSE54
acos()	4.5.2.1	MAN	MAN	MAN	MAN
asin()	4.5.2.2	MAN	MAN	MAN	MAN
atan()	4.5.2.3	MAN	MAN	MAN	MAN
atan2()	4.5.2.4	MAN	MAN	MAN	MAN
ceil()	4.5.6.1	MAN	MAN	MAN	MAN
cos()	4.5.2.5	MAN	MAN	MAN	MAN
cosh()	4.5.3.1	MAN	MAN	MAN	MAN
exp()	4.5.4.1	MAN	MAN	MAN	MAN
fabs()	4.5.6.2	MAN	MAN	MAN	MAN
floor()	4.5.6.3	MAN	MAN	MAN	MAN
fmod()	4.5.6.4	MAN	MAN	MAN	MAN
frexp()	4.5.4.2	MAN	MAN	MAN	MAN
ldexp()	4.5.4.3	MAN	MAN	MAN	MAN
log()	4.5.4.4	MAN	MAN	MAN	MAN
log10()	4.5.4.5	MAN	MAN	MAN	MAN
modf()	4.5.4.6	MAN	MAN	MAN	MAN
pow()	4.5.5.1	MAN	MAN	MAN	MAN
sin()	4.5.2.6	MAN	MAN	MAN	MAN
sinh()	4.5.3.2	MAN	MAN	MAN	MAN
sqrt()	4.5.5.2	MAN	MAN	MAN	MAN
tan()	4.5.2.7	MAN	MAN	MAN	MAN
tanh()	4.5.3.3	MAN	MAN	MAN	MAN

NOTE:

MAN — Mandatory for this profile.

Table A-19 – POSIX_C_LANG_SUPPORT Non-Local Jump Functions

Function	Reference in the C Standard	Status			
		PSE51	PSE52	PSE53	PSE54
longjmp()	4.6.2.1	MAN	MAN	MAN	MAN
setjmp()	4.6.1.1	MAN	MAN	MAN	MAN

NOTE:

MAN — Mandatory for this profile.

Table A-20 – POSIX_C_LANG_SUPPORT General Functions

Function	Reference in the C Standard	Status			
		PSE51	PSE52	PSE53	PSE54
abs()	4.10.6.1	MAN	MAN	MAN	MAN
atof()	4.10.1.1	MAN	MAN	MAN	MAN
atoi()	4.10.1.2	MAN	MAN	MAN	MAN
atol()	4.10.1.3	MAN	MAN	MAN	MAN
bsearch()	4.10.5.1	MAN	MAN	MAN	MAN
calloc()	4.10.3.1	MAN	MAN	MAN	MAN
free()	4.10.3.2	MAN	MAN	MAN	MAN
malloc()	4.10.3.3	MAN	MAN	MAN	MAN
qsort()	4.10.5.2	MAN	MAN	MAN	MAN
rand()	4.10.2.1	MAN	MAN	MAN	MAN
realloc()	4.10.3.4	MAN	MAN	MAN	MAN
srand()	4.10.2.2	MAN	MAN	MAN	MAN

NOTE:

MAN — Mandatory for this profile.

Table A-21 – POSIX_C_LANG_SUPPORT String Handling Functions

Function	Reference in the C Standard	Status			
		PSE51	PSE52	PSE53	PSE54
strcat()	4.11.3.1	MAN	MAN	MAN	MAN
strchr()	4.11.5.2	MAN	MAN	MAN	MAN
strcmp()	4.11.4.2	MAN	MAN	MAN	MAN
strcpy()	4.11.2.3	MAN	MAN	MAN	MAN
strcspn()	4.11.5.3	MAN	MAN	MAN	MAN
strlen()	4.11.6.3	MAN	MAN	MAN	MAN
strncpy()	4.11.2.4	MAN	MAN	MAN	MAN
strncat()	4.11.3.2	MAN	MAN	MAN	MAN
strncmp()	4.11.4.4	MAN	MAN	MAN	MAN
strpbkr()	4.11.5.4	MAN	MAN	MAN	MAN
strrchr()	4.11.5.5	MAN	MAN	MAN	MAN
strspn()	4.11.5.6	MAN	MAN	MAN	MAN
strstr()	4.11.5.7	MAN	MAN	MAN	MAN
strtok()	4.11.5.8	MAN	MAN	MAN	MAN

NOTE:

MAN — Mandatory for this profile.

Table A-22 – POSIX_C_LANG_SUPPORT Date and Time Functions

Function	Reference in the C Standard	Status			
		PSE51	PSE52	PSE53	PSE54
asctime()	4.12.3.1	MAN	MAN	MAN	MAN
ctime()	4.12.3.2	MAN	MAN	MAN	MAN
gmtime()	4.12.3.3	MAN	MAN	MAN	MAN
localtime()	4.12.3.4	MAN	MAN	MAN	MAN
mktime()	4.12.2.3	MAN	MAN	MAN	MAN
strftime()	4.12.3.5	MAN	MAN	MAN	MAN
time()	4.12.2.4	MAN	MAN	MAN	MAN
tzset()	4.12.2.4	MAN	MAN	MAN	MAN

NOTE:

MAN — Mandatory for this profile.

Rights Reserved

A.3.2 POSIX.1b

Table A-23 contains the required options, limits, and any other constraints on POSIX.1b.

Table A-23 – POSIX.1b Option Requirements

Option	Profile			
	PSE51	PSE52	PSE53	PSE54
{_POSIX_ASYNCHRONOUS_IO}	NRQ	MAN	MAN	MAN
{_POSIX_MAPPED_FILES}	NRQ	MAN	NRQ	MAN
{_POSIX_MEMLOCK}	MAN	MAN	MAN	MAN
{_POSIX_MEMLOCK_RANGE}	MAN	MAN	MAN	MAN
{_POSIX_MEMORY_PROTECTION}	NRQ	NRQ	MAN	MAN
{_POSIX_MESSAGE_PASSING}	MAN	MAN	MAN	MAN
{_POSIX_PRIORITIZED_IO}	NRQ	NRQ	MAN	MAN
{_POSIX_PRIORITY_SCHEDULING}	NRQ	NRQ	MAN	MAN
{_POSIX_REALTIME_SIGNALS}	MAN	MAN	MAN	MAN
{_POSIX_SEMAPHORES}	MAN	MAN	MAN	MAN
{_POSIX_SHARED_MEMORY_OBJECTS}	MAN	MAN	MAN	MAN
{_POSIX_SYNCHRONIZED_IO}	MAN	MAN	MAN	MAN
{_POSIX_TIMERS}	MAN	MAN	MAN	MAN
{_POSIX_FSYNC}	MAN	MAN	MAN	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.3 POSIX.1c

Tables A-24 through A-29 contain the required options, limits, and any other constraints on POSIX.1c.

Table A-24 – POSIX.1c Option Requirements

Option	Profile			
	PSE51	PSE52	PSE53	PSE54
{_POSIX_THREADS}	MAN	MAN	MAN	MAN
{_POSIX_THREAD_ATTR_STACKADDR}	MAN	MAN	MAN	MAN
{_POSIX_THREAD_ATTR_STACKSIZE}	MAN	MAN	MAN	MAN
{_POSIX_THREAD_PRIO_INHERIT}	MAN	MAN	MAN	MAN
{_POSIX_THREAD_PRIO_PROTECT}	MAN	MAN	MAN	MAN
{_POSIX_THREAD_PRIORITY_SCHEDULING}	MAN	MAN	MAN	MAN
{_POSIX_THREAD_PROCESS_SHARED}	NRQ	NRQ	MAN	MAN
{_POSIX_THREAD_SAFE_FUNCTIONS}	PRT	PRT	PRT	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

PRT — Partial, only the subset of units of functionality called out in A.3.3.

A.3.3.1 Reentrant User Group Function Behavior

The function listed in Table A-25 shall behave as described in the referenced clause.

Table A-25 – POSIX_USER_GROUPS_R Function

Function	Reference in POSIX.1c	Status			
		PSE51	PSE52	PSE53	PSE54
getlogin_r()	4.2.4	NRQ	NRQ	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.3.2 Reentrant Device-Specific Function Behavior

The function listed in Table A-26 shall behave as described in the referenced clause.

Table A-26 – POSIX_DEVICE_SPECIFIC_R Function

Function	Reference in POSIX.1c	Status			
		PSE51	PSE52	PSE53	PSE54
ttyname_r()	4.7.4	NRQ	NRQ	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.3.3 File Locking Function Behavior

The functions listed in Table A-27 shall behave as described in the referenced clause.

Table A-27 – POSIX_FILE_LOCKING Functions

Function	Reference in POSIX.1c	Status			
		PSE51	PSE52	PSE53	PSE54
getc_unlocked()	8.2.7	MAN	MAN	MAN	MAN
getchar_unlocked()	8.2.7	MAN	MAN	MAN	MAN
flockfile()	8.2.6	MAN	MAN	MAN	MAN
ftrylockfile()	8.2.6	MAN	MAN	MAN	MAN
funlockfile()	8.2.6	MAN	MAN	MAN	MAN
putc_unlocked()	8.2.7	MAN	MAN	MAN	MAN
putchar_unlocked()	8.2.7	MAN	MAN	MAN	MAN

NOTE:

MAN — Mandatory for this profile.

A.3.3.4 Reentrant C Language Support Function Behavior

The functions listed in Table A-28 shall behave as described in the referenced clause.

Table A-28 – POSIX_C_LANG_SUPPORT_R Functions

Function	Reference in POSIX.1c	Status PSE51	PSE52	PSE53	PSE54
asctime_r()	8.3.5	MAN	MAN	MAN	MAN
ctime_r()	8.3.6	MAN	MAN	MAN	MAN
gmtime_r()	8.3.7	MAN	MAN	MAN	MAN
localtime_r()	8.3.8	MAN	MAN	MAN	MAN
rand_r()	8.3.9	MAN	MAN	MAN	MAN
strtok_r()	8.3.4	MAN	MAN	MAN	MAN

NOTE:

MAN — Mandatory for this profile.

A.3.3.5 Reentrant System Database Function Behavior

The functions listed in Table A-29 shall behave as described in the referenced clause.

Table A-29 – POSIX_SYSTEM_DATABASE_R Functions

Function	Reference in POSIX.1c	Status PSE51	PSE52	PSE53	PSE54
getgrgid_r()	9.2.1	NRQ	NRQ	NRQ	MAN
getgrnam_r()	9.2.1	NRQ	NRQ	NRQ	MAN
getpwnam_r()	9.2.2	NRQ	NRQ	NRQ	MAN
getwuid_r()	9.2.2	NRQ	NRQ	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

A.3.4 POSIX.2/2a

Table A-30 contains the required options, limits, and any other constraints on POSIX.2/2a.

Table A-30 – POSIX.2/2a Option Requirements

Option	Profile			
	PSE51	PSE52	PSE53	PSE54
{POSIX2_C_BIND}	DEV	DEV	DEV	MAN
{POSIX2_C_DEV}	DEV	DEV	DEV	MAN
{POSIX2_CHAR_TERM}	NRQ	NRQ	NRQ	MAN
{POSIX2_FORT_DEV}	NRQ	NRQ	NRQ	NRQ
{POSIX2_FORT_RUN}	NRQ	NRQ	NRQ	MAN
{POSIX2_LOCALEDEF}	NRQ	NRQ	NRQ	OPT
{POSIX2_SW_DEV}	DEV	DEV	DEV	MAN
{POSIX2_UPE}	NRQ	NRQ	NRQ	MAN

NOTE:

DEV — Required in development environment for this profile; not required in the execution environment for this profile.

MAN — Mandatory for this profile.

NRQ — Not required for this profile.

OPT — Optional for this profile.

Annex B
(normative)

ISPICS Requirements List (Ada)

B.1 Options

This document defines four different profile options that should cover most application environments within the realtime class. These profile options are listed in Table B-1.

Table B-1 – Profile Options (Ada)

Profile	Ada Identifier	Status
PSE51	POSIX_Profiles.Realtime_Minimal	OPT
PSE52	POSIX_Profiles.Realtime_Controller	OPT
PSE53	POSIX_Profiles.Realtime_Dedicated	OPT
PSE54	POSIX_Profiles.Realtime_Multi	OPT

NOTE:

OPT — Optional, one or more profiles may be supported.

The package POSIX_Profiles shall be supported by all profiles. The Boolean subtypes contained in this package shall indicate the profiles and options supported by the implementation. Supported profiles and options shall be indicated by the appropriate indentifier having the range True..True; unsupported profiles and options shall have the range False..False.

```
Package POSIX_Profiles is

  -- profile options
  subtype Realtime_Minimal    is Boolean range <Implementation Defined>;
  subtype Realtime_Controller is Boolean range <Implementation Defined>;
  subtype Realtime_Dedicated  is Boolean range <Implementation Defined>;
  subtype Realtime_Multi      is Boolean range <Implementation Defined>;

  -- Language Development Options
  subtype Realtime_Lang_c89   is Boolean range <Implementation Defined>;
  subtype Realtime_Lang_Ada95 is Boolean range <Implementation Defined>;
  subtype Realtime_Lang_F77 is Boolean range <Implementation Defined>;

End POSIX_Profiles;
```

B.2 Standards

The following standards are required in whole or in part by one or more of the four included profiles.

Table B-2 – Ada Standards

Standard	PSE51	PSE52	PSE53	PSE54
POSIX.5b	PRT	PRT	PRT	ALL
Ada RM	PRT	PRT	PRT	PRT
Cstd + POSIX.1, .1b, .1c	OPT	OPT	OPT	OPT
POSIX.2/2a	PRT	PRT	PRT	PRT

NOTE:

ALL — Required for all four profiles.

PRT — Partial, only the subset options called out in B.3.

OPT — Optional for this profile.

B.3 Constraints

Some of the realtime profiles defined in this standard require only specific Units of Functionality of the required standards. The absence of particular elements of these standards introduces constraints on the use of some of the features of particular functions. This subclause defines the constraints that an application strictly conforming to one of the profiles shall observe when using each of the functions required by that profile.

B.3.1 POSIX.5b

The following tables contain the required options, limits, and any other constraints on POSIX.5b.

Table B-3 – POSIX.5b Limits Requirements

Limits	PSE51	PSE52	PSE53	PSE54
POSIX_Limits.Groups_Maxima'First	0	0	0	>=8
POSIX_Limits.Change_Owner_Restriction	NRQ	NRQ	NRQ	MAN
POSIX_Limits.Job_Control_Support	NRQ	NRQ	NRQ	MAN
POSIX_Limits.Filename_Truncation	NAM	PRI	PRI	PRI
POSIX_Limits.Saved_IDs_Support	NRQ	NRQ	NRQ	MAN
POSIX_Limits.Disable_Control_Character	NRQ	NRQ	NRQ	MAN
POSIX_Limits.Signal_Entries_Support	NRQ	NRQ	NRQ	NRQ

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

NAM — Mandatory to generate an error for object names longer than NAME_MAX.

PRI — The primary file system shall generate an error for pathname components longer than NAME_MAX. The user is responsible for semantics of other file systems that may be mounted.

The ranges of the following subtypes in the package POSIX_Options shall be False..False in PSE51, PSE52, and PSE53. For PSE54, they shall have the range True..True:

Job_Control_Support.

Saved_IDs_Support.

Change_Owner_Restriction.

The range of the subtype POSIX_Options.Filename_Truncation shall be False..False in all four profiles.

In all profiles that do not support the POSIX_JOB_CONTROL Unit of Functionality, the subprogram POSIX_Signals.Set_Stopped_Child_Signal shall fail silently.

In all profiles that do not support the POSIX_JOB_CONTROL Unit of Functionality, the subprogram POSIX_Signals.Stopped_Child_Signal_Enabled shall return False.

POSIX_Limits.Groups_Maxima'First shall be zero for PSE51, PSE52, and PSE53. For PSE54 it shall be greater than or equal to eight.

POSIX_Terminal_Functions.Disable_Control_Character (which corresponds to _POSIX_VDISABLE) is not supported in PSE51, PSE52, and PSE53. For PSE54, POSIX_Terminal_Functions.Disable_Control_Character shall not raise POSIX_Error with an error code of Operation_Not_Implemented.

For PSE51 and PSE52, the blocking behavior of all operations defined by POSIX.5b shall be Tasks (see POSIX.5b, clause 2.4.1.5).

Subprograms not supported by a given profile shall raise POSIX_Error, returning an error code of Operation_Not_Supported, except as noted otherwise.

All `Image` and `Value` functions that appear in the packages supported by a profile must be implemented.

Where an overloaded subprogram is required by a Unit of Functionality, all forms of the subprogram appearing in the referenced clause must be supported, except as otherwise noted.

B.3.1.1 Single Process Function Behavior

The subprograms in Table B-4 shall behave as described in the referenced clause, except that:

(1) An application strictly conforming to PSE51, PSE52, PSE53, or PSE54 shall not call the functions `POSIX_Configurable_System_-Limits.System_POSIX_Version` or `POSIX_Configurable_System_-Limits.System_POSIX_Ada_Version`, since a meaningful value cannot be returned.

(2) A conforming application must act as if `POSIX_Limits.Child_-Processes_Maxima'Last` = 0.

Table B-4 – POSIX_SINGLE_PROCESS Subprograms

Package/Subprogram	Reference in POSIX.5b	Status			
		PSE51	PSE52	PSE53	PSE54
POSIX: All	2.4	MAN	MAN	MAN	MAN
POSIX_Limits: All	2.6	MAN	MAN	MAN	MAN
POSIX_Options: All	2.5	MAN	MAN	MAN	MAN
POSIX_Configurable_System_Limits: All	4.5	MAN	MAN	MAN	MAN
POSIX_Calendar: All	4.4	MAN	MAN	MAN	MAN
POSIX_Process_Environment:	4.3	MAN	MAN	MAN	MAN
Argument_List	4.3.1				
Copy_From_Current_Environment	4.3.2				
Copy_To_Current_Environment	4.3.2				
Copy_Environment	4.3.2				
Clear_Environment	4.3.2				
Set_Environment_Variable	4.3.2				
Delete_Environment_Variable	4.3.2				
Length	4.3.2				
For_Every_Environment_Variable	4.3.2				
For_Every_Current_Environment_Variable	4.3.2				
Environment_Value_Of	4.3.2				
Is_Environment_Variable	4.3.2				

NOTE:

MAN — Mandatory for this profile.

B.3.1.2 Multi-Process Function Behavior

The subprograms listed in Table B-5 shall behave as described in the referenced clause, except as follows :

(1) An application strictly conforming to PSE53 shall use the **Pathname** parameter of **POSIX_Process_Primitives.Start_Process**, or the **Filename** parameter of **POSIX_Process_Primitives.Start_Process_Search** only to specify the name of the process image to be executed, without any file system semantics implied, because this profile does not require general file system capabilities.

(2) An application strictly conforming to PSE53 shall not call the subprograms contained in the package **POSIX_Unsafe_Process_Primitives**, but shall instead rely upon either **POSIX_Process_Primitives.Start_Process** or **POSIX_Process_Primitives.Start_Process_Search** to create new processes.

Table B-5 – POSIX_MULTI_PROCESS Subprograms

Package/Subprogram	Reference in POSIX.5b	Status			
		PSE51	PSE52	PSE53	PSE54
POSIX_Process_Primitives: All	3.1	NRQ	NRQ	MAN	MAN
POSIX_Unsafe_Process_Primitives: All	3.2	NRQ	NRQ	NRQ	MAN
POSIX_Process_Times: All	4.2	NRQ	NRQ	MAN	MAN
POSIX_Process_Identification: Get_Process_ID Get_Parent_Process_ID	4.1 4.1 4.1	NRQ	NRQ	MAN	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

B.3.1.3 Job Control Function Behavior

The functions listed in Table B-6 shall behave as described in the referenced clause with the following additional constraints:

(1) In all profiles that do not support the POSIX_JOB_CONTROL Unit of Functionality, the subprogram POSIX_Signals.Set_Stopped_Child_Signal shall fail silently.

(2) In all profiles that do not support the POSIX_JOB_CONTROL Unit of Functionality, the subprogram POSIX_Signals.Stopped_Child_Signal_Enabled shall return False.

Table B-6 – POSIX_JOB_CONTROL Subprograms

Package/Subprogram	Reference in POSIX.5b	Status			
		PSE51	PSE52	PSE53	PSE54
POSIX_Process_Identification: Set_Process_Group_ID Create_Process_Group	4.1 4.1.2 4.1.2	NRQ	NRQ	NRQ	MAN
POSIX_Terminal_Functions: Get_Process_Group_ID Set_Process_Group_ID	7.2 7.2.11 7.2.11	NRQ	NRQ	NRQ	MAN
POSIX_Signals: Set_Stopped_Child_Signal Stopped_Child_Signal_Enabled	3.3 3.3.10 3.3.10	NRQ	NRQ	NRQ	MAN

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

B.3.1.4 Signals Subprograms Behavior

The subprograms listed in Table B-7 shall behave as described in the referenced clause, except for the following constraints :

(1) An application strictly conforming to PSE52 or PSE53 shall be considered erroneous if any signal results in abnormal termination of the process because these profiles do not support multiple processes.

(2) An application strictly conforming to PSE52, PSE53, or PSE54 shall not call the form of POSIX_Signals.Send_Signal that takes a process group ID as an argument because these profiles do not require process group functionality. An application strictly conforming to PSE51, PSE52, PSE53, or PSE54 shall not attempt to bind a signal to a task entry.

Table B-7 – POSIX_SIGNALS Subprograms

Package/Subprogram	Reference in POSIX.5b	Status			
		PSE51	PSE52	PSE53	PSE54
POSIX_Signals:	3.3	MAN	MAN	MAN	MAN
Add_Signal	3.3.7				
Add_All_Signals	3.3.7				
Delete_Signal	3.3.7				
Delete_All_Signals	3.3.7				
Is_Member	3.3.7				
Send_Signal	3.3.18				
Set_Blocked_Signals	3.3.8				
Block_Signals	3.3.8				
Unblock_Signals	3.3.8				
Blocked_Signals	3.3.8				
Ignore_Signal	3.3.9				
Unignore_Signal	3.3.9				
Is_Ignored	3.3.9				
Pending_Signals	3.3.11				
Await_Signal*	3.3.15				
Await_Signal_Or_Timeout*	3.3.15				
Interrupt_Task	3.3.20				
Get_Signal**	3.3.12				
Set_Signal**	3.3.12				
Get_Notification	3.3.12				
Set_Notification	3.3.12				
Get_Data**	3.3.12				
Set_Data**	3.3.12				

NOTE:

MAN — Mandatory for this profile.

* — (Return type Signal)

** — (Operation type Signal_Event)

B.3.1.5 User Group Subprogram Behavior

The subprograms listed in Table B-8 shall behave as described in the referenced clause.

Table B-8 – POSIX_USER_GROUPS Subprograms

Package/Subprogram	Reference	Status			
	in POSIX.5b	PSE51	PSE52	PSE53	PSE54
POSIX_Process_Identification:	4.1	NRQ	NRQ	NRQ	MAN
Get_Real_User_ID	4.1.3				
Get_Effective_User_ID	4.1.3				
Get_Real_Group_ID	4.1.4				
Get_Effective_Group_ID	4.1.4				
Set_User_ID	4.1.3				
Create_Session	4.1.2				
Set_Group_ID	4.1.4				
Get_Groups	4.1.4				
Get_Login_Name	4.1.3				
Get_Process_Group_ID	4.1.3				

NOTE:

NRQ — Not required for this profile.

MAN — Mandatory for this profile.

B.3.1.6 File System Subprogram Behavior

The subprograms listed in Table B-9 shall behave as described in the referenced clause.