INTERNATIONAL STANDARD

ISO/IEC 14515-1

First edition
2000-12-15

AMENDMENT 1
2003-12-15

IEEE Std 2003.1b-2000
(Amendment to
IEEE Std 2003.1-1992)

Information technology — Portable Operating System Interface (POSIX®) — Test methods for measuring conformance to POSIX —

Part 1:
System interfaces

AMENDMENT 1: Realtime Extension
(C Language)

Technologies de l'information — Interface de système de fonctionnement portable (POSIX®) — Méthodes d'essai pour mesurer la conformité au POSIX —

Partie 1: Interfaces de système

AMENDEMENT 1: Extension en temps réel (langage C)

Reference number
ISO/IEC 14515-1:2000/Amd.1:2003(E)
IEEE Std 2003.1b-2000
(Amendment to IEEE Std 2003.1-1992)

AVA Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

ISO/IEC 14515-1:2000/Amd.1:2003(E)
IEEE Std 2003.1b-2000
(Amendment to IEEE Std 2003.1-1992)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

ISO

Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org

Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

ISO/IEC 14515-1:2000/Amd.1:2003(E)
IEEE Std 2003.1b™-2000
(Amendment to IEEE Std 2003.1™-1992)

Information Technology—Portable Operating System Interface (POSIX®)—Part 1: Test method for measuring conformance to POSIX

Amendment 1: Realtime Extension (C Language)

Sponsor

Portable Applications Standards Committee
of the
IEEE Computer Society

Approved 30 March 2000
IEEE-SA Standards Board

Approved 2 November 2000
American National Standards Institute

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.

Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Abstract: This standard defines the test method specifications for IEEE Std 1003.b-1993 (based on the document corresponding to the merger of IEEE Std 1003.1-1990 and IEEE Std 1003.1b-1993). The test method specifications consist of assertions to be tested and related test procedures. As an amendment to IEEE Std 1003.1-1990, this standard is structured to amend those portions of IEEE Std 2003.1-1992 (the test method specification for IEEE Std 1003.1-1990) that correspond to the amended parts of IEEE Std 1003.1-1990. This standard is aimed primarily at providers of test methods for IEEE Std 1003.1b-1993 and at implementors of IEEE Std 1003.1b-1993.

Keywords: assertion, assertion test, C programming language, POSIX, POSIX Conformance Document, POSIX Conformance Test Procedure, POSIX Conformance Test Suite, realtime, test method specification, test result code

---

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

Copyright © 2003 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 15 August 2003. Printed in the United States of America.

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by the Institute of Electrical and Electronics Engineers, Inc.

POSIX is a registered trademark of the Institute of Electrical and Electronics Engineers, Inc.

Print: ISBN 0-7381-3759-6 SH95152
PDF: ISBN 0-7381-3760-X SS95152

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

International Standard ISO/IEC 14515-1:2000/Amd.1:2003(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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard 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 document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Amendment 1 to ISO/IEC 14515-1:2000 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*.

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

Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

Use of an IEEE Standard is wholly voluntary. The IEEE disclaims liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEEE Standard document.

The IEEE does not warrant or represent the accuracy or content of the material contained herein, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEEE Standards documents are supplied "**AS IS**."

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.

In publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

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 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 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. 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.

Authorization to photocopy portions of any individual standard for internal or personal use is granted by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Contents

Introduction viii
Organization of This Standard viii
How to Read This Standard viii
Related Standards Activities ix

Section 1: General 1
1.1 Scope 1
1.2 Normative References 2
1.3 Conformance 3
1.4 Test Methods 4

Section 2: Terminology and General Requirements 19
2.1 Conventions 19
2.2 Definitions 20
2.3 General Concepts 28
2.4 Error Numbers 34
2.5 Primitive System Data Types 35
2.6 Environment Description 35
2.7 C Language Definitions 37
2.8 Numerical Limits 40
2.9 Symbolic Constants 43

Section 3: Process Primitives 48
3.1 Process Creation and Execution 48
3.1.1 Process Creation 48
3.1.2 Execute a File 50
3.2 Process Termination 53
3.2.1 Wait for Process Termination 53
3.2.2 Terminate a Process 53
3.3 Signals 55
3.3.1 Signal Concepts 55
3.3.2 Send a Signal to a Process 65
3.3.3 Manipulate Signal Sets 65
3.3.4 Examine and Change Signal Action 65
3.3.5 Examine and Change Blocked Signals 67
3.3.6 Examine Pending Signals 67
3.3.7 Wait for a Signal 68
3.3.8 Synchronously Accept a Signal 68
3.3.9 Queue a Signal to a Process 72
3.4 Timer Operations 75
3.4.1 Schedule Alarm 76
3.4.2 Suspend Process Execution 76
3.4.3 Delay Process Execution 76

Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Section 4: Process Environment 78
4.8 Configurable System Variables 78
4.8.1 Get Configurable System Variables 78

Section 5: Files and Directories 82
5.1 Directories 82
5.1.1 Format of Directory Entries 82
5.1.2 Directory Operations 82
5.2 Working Directory 82
5.2.1 Change Current Working Directory 82
5.2.2 Get Working Directory Pathname 83
5.3 General File Creation 83
5.3.1 Open a File 83
5.3.2 Create New File or Rewrite an Existing One 84
5.3.3 Set File Creation Mask 84
5.3.4 Link to a File 85
5.4 Special File Creation 85
5.4.1 Make a Directory 85
5.4.2 Make a FIFO Special File 85
5.5 File Removal 86
5.5.1 Remove Directory Entries 86
5.5.2 Remove a Directory 86
5.5.3 Rename a File 86
5.6 File Characteristics 87
5.6.1 File Characteristics: Header and Data Structure 87
5.6.2 Get File Status 88
5.6.3 Check File Accessibility 89
5.6.4 Change File Modes 89
5.6.5 Change Owner and Group of a File 92
5.6.6 Set File Access and Modification Times 93
5.6.7 Truncate a File to a Specified Length 93
5.7 Configurable Pathname Variables 96
5.7.1 Get Configurable Pathname Variables 96

Section 6: Input and Output Primitives 99
6.1 Pipes 99
6.1.1 Create an Inter-Process Channel 99
6.2 File Descriptor Manipulation 99
6.2.1 Duplicate an Open File Descriptor 99
6.3 File Descriptor Deassignment 100
6.3.1 Close a File 100
6.4 Input and Output 101
6.4.1 Read from a File 101
6.4.2 Write to a File 102
6.5 Control Operations on Files 103
6.5.1 Data Definitions for File Control Operations 103
6.5.2 File Control 103
6.5.3 Reposition Read/Write File Offset 106
6.6 File Synchronization 107

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

6.6.1 Synchronize the State of a File 107
6.6.2 Synchronize the Data of a File 110
6.7 Asynchronous Input and Output 112
6.7.1 Data Definitions for Asynchronous Input and Output 112
6.7.2 Asynchronous Read 115
6.7.3 Asynchronous Write 121
6.7.4 List Directed I/O 128
6.7.5 Retrieve Error Status of Asynchronous I/O Operation 138
6.7.6 Retrieve Return Status of Asynchronous I/O Operation 140
6.7.7 Cancel Asynchronous I/O Request 141
6.7.8 Wait for Asynchronous I/O Request 144
6.7.9 Asynchronous File Synchronization 147

Section 7: Device- and Class- Specific Functions 153

Section 8: Language-Specific Services for the C Programming Language 154
8.2.2 Open a Stream on a File Descriptor 154

Section 9: System Databases 155

Section 10: Data Interchange Format 156

Section 11: Synchronization 157
11.1 Semaphore Characteristics 157
11.2 Semaphore Functions 158
11.2.1 Initialize an Unnamed Semaphore 158
11.2.2 Destroy an Unnamed Semaphore 161
11.2.3 Initialize/Open a Named Semaphore 162
11.2.4 Close a Named Semaphore 168
11.2.5 Remove a Named Semaphore 170
11.2.6 Lock a Semaphore 172
11.2.7 Unlock a Semaphore 180
11.2.8 Get the Value of a Semaphore 184

Section 12: Memory Management 187
12.1 Memory Locking Functions 189
12.1.1 Lock/Unlock the Address Space of a Process 189
12.1.2 Lock/Unlock a Range of Process Address Space 193
12.2 Memory Mapping Functions 197
12.2.1 Map Process Address to a Memory Object 197
12.2.3 Change Memory Protection 207
12.2.4 Memory Object Synchronization 211
12.3 Shared Memory Functions 215
12.3.1 Open a Shared Memory Object 215
12.3.2 Remove a Shared Memory Object 222

Section 13: Execution Scheduling 225
13.1 Scheduling Parameters 225

Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

13.2 Scheduling Policies 225
13.2.1 SCHED_FIFO 226
13.2.2 SCHED_RR 228
13.2.3 SCHED_OTHER 228
13.3 Process Scheduling Functions 229
13.3.1 Set Scheduling Parameters 229
13.3.2 Get Scheduling Parameters 232
13.3.3 Set Scheduling Policy and Scheduling Parameters 234
13.3.4 Get Scheduling Policy 238
13.3.5 Yield Processor 239
13.3.6 Get Scheduling Parameter Limits 240

Section 14: Clocks and Timers 245
14.1 Data Definitions for Clocks and Timers 245
14.1.1 Time Value Specification Structures 245
14.1.2 Timer Event Notification Control Block 246
14.1.3 Type Definitions 246
14.1.4 Manifest Constants 246
14.2 Clocks and Timer Functions 247
14.2.1 Clocks 247
14.2.2 Create a Per-Process Timer 253
14.2.3 Delete a Per-Process Timer 257
14.2.4 Per-Process Timers 259
14.2.5 High Resolution Sleep 265

Section 15: Message Passing 269
15.1 Data Definitions for Message Queues 269
15.1.1 Data Structures 269
15.2 Message Passing Functions 270
15.2.1 Open a Message Queue 270
15.2.2 Close a Message Queue 278
15.2.3 Remove a Message Queue 280
15.2.4 Send a Message to a Message Queue 282
15.2.5 Receive a Message From a Message Queue 285
15.2.6 Notify Process that a Message is Available on a Queue 287
15.2.7 Set Message Queue Attributes 289
15.2.8 Get Message Queue Attributes 291

Annex A
(normative)

Conforming Test Results 293
A.1 General 293
A.2 Terminology and General Requirements 293
A.3 Process Primitives 295
A.4 Process Environment 302
A.5 Files and Directories 302
A.6 Input and Output Primitives 306
A.7 Device- and Class-Specific Functions 314
A.8 Language-Specific Services for the C Programming Language 314

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

A.9 System Databases 314
A.10 Data Interchange Format 314
A.11 Synchronization 314
A.12 Memory Management 319
A.13 Execution Scheduling 326
A.14 Clocks and Timers 330
A.15 Message Passing 335

Annex B
(informative)

Bibliography 341
B.1 Related Open Systems Standards 341
B.2 Other Standards 343
B.3 Historical Documentation and Introductory Texts 343
B.4 Other Sources of Information 345

Identifier Index 347

Alphabetic Topical Index 350

TABLES

Table 1-1 – PCTS Symbols and Values 6
Table 2-1 – Minimum Values 40
Table 2-2 – Run-Time Invariant Values (Possibly Indeterminate) 42
Table 2-3 – Maximum Values 43
Table 2-4 – Compile-Time Symbolic Constants 44
Table 2-5 – Execution-Time Symbolic Constants 46
Table 3-1 – Required Signals 55
Table 3-2 – Job Control Signals 56
Table 3-3 – Memory Protection Signals 56
Table 4-2 – Configurable System Variables 80

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Introduction

(This Introduction is not a normative part of IEEE Std 2003.1b-2000, IEEE Standard for Information Technology–Test Methods Specifications for Measuring Conformance to POSIX-Part 1: System Application Program Interface(API)-Amendment 1: Realtime Extension [C Language]).

This standard defines the test method specifications for IEEE Std 1003.1b-1993 (based on the document corresponding to the merger of IEEE Std 1003.1-1990 and IEEE Std 1003.1b-1993). The test method specifications consist of assertions to be tested and related test procedures. As an amendment to IEEE Std 1003.1-1990, this standard is structured to amend those portions of IEEE Std. 2003.1-1992 {4} (the test method specification for IEEE Std 1003.1-1990) that correspond to the amended parts of IEEE Std 1003.1-1990. This standard is aimed primarily at providers of test methods for IEEE Std 1003.1b-1993 and at implementors of IEEE Std 1003.1b-1993.

Organization of This Standard

This document is organized into five parts, as follows:

(1) Statement of scope, normative references, conformance requirements, and test methods (Section 1)

(2) Conventions and definitions (Section 2)

(3) Assertions to test POSIX.1b {3} (Sections 2 through 15)

(4) Conforming test results (Annex A)

(5) Bibliography (Annex B)

This introduction, any footnotes, notes accompanying the test, and the *informative* annexes are not considered part of this standard. Annex A is normative. Annex B is informative.

How to Read This Standard

This document is organized using the same section numbering as POSIX.1b {3}, in order to facilitate finding the testing requirements for a particular element of POSIX.1b {3}. Subclause 1.4 has been added to this document to keep the section numbering consistent with POSIX.1b {3} and to provide a place to describe features relevant to the test methods. It is strongly recommended that the reader review 1.4 after reading 1.1, 1.2, and this introduction. Where possible, this document tries to use the same assertion numbering that was used as IEEE Std 2003.1-1992 {4}. This document skips over the assertion numbers where there are no assertions that correspond to an assertion in IEEE Std 2003.1-1992 {4}. The assertions have been numbered with a fractional part where more than one assertion takes the place of a single assertion in IEEE Std 2003.1-1992 {4}.

Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Related Standards Activities

Activities to extend this standard 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: [1)]

(1) Language-independent service descriptions of this standard

(2) C, Ada, and FORTRAN language bindings to (1)

(3) Shell and utility facilities

(4) Verification testing methods

(5) Multithreaded process 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; transaction processing; realtime systems; and multiuser systems based on historical models

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

Extensions are approved as "amendments" or "revisions" to this document, following IEEEStandards procedures.

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

If you have an 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 an interest in participating in this work at the international level, contact your ISO/IEC national body.

---

1)

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.

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

IEEE Std 2003.1b-2000 was prepared by a breakout group in the 2003 Working Group focused on developing the IEEE 2003.1b standard, sponsored by the Portable Applications Standards Committee of the IEEE Computer Society. At the time this standard was approved, the membership of the IEEE 2003.1b breakout groups was as follows:

Portable Applications Standards Committee

Chair:	Lowell Johnson
Vice Chair:	Joe Gwinn
Secretary:	Nick Stoughton
Functional Chairs:	Andrew Josey
	Jay Ashford
	Curtis Royster
	Jason Zions

2003 Working Group Officials

Chair:	Barry Hedquist
	Roger Martin (retired)
Vice Chair:	John Davies (2003)
	Ken Thomas (2003.1b)
Editors:	Bruce Weiner (primary)
	Jeffrey S. Haemer (supporting)
	Barry Hedquist
Secretary:	Keith Stobie (1994)

Technical Reviewers

Ted Baker	John Davies	Barry Hedquist
Jeffrey Haemer	Ken Thomas	Bruce Weiner

2003.1b Working Group

John Davies	Leo Hansen	Ken Thomas
Jeffrey Haemer	Curtis Royster	Bruce Weiner

The following members of the balloting committee voted on this standard:

Khaled Al-Ali	Barry Hedquist	William R. Smith
Andy Bihain	Lowell Johnson	Kenneth G. Thomas
Susan Corwin	Roger Martin	Bruce Weiner
Steven J. Dovitch	Gerald Powell	Andrew E. Wheeler
Michel P. Gien	Curtis Royster	Alex White
Patrick Hebert		John Zolnowsky

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

When the IEEE-SA Standards Board approved this standard on 30 March 2000, it had the following membership:

Donald N. Heirman, *Chair*
James T. Carlo, *Vice Chair*
Judith Gorman, *Secretary*

Satish K. Aggarwal
Mark D. Bowman
Gary R. Engmann
Harold E. Epstein
H. Landis Floyd
Jay Forster*
Howard M. Frazier
Ruben D. Garzon
James H. Gurney
Richard J. Holleman
Lowell G. Johnson
Robert J. Kennelly
Joseph L. Koepfinger*
Peter H. Lips
L. Bruce McClung
Daleep C. Mohla
James W. Moore
Robert F. Munzner
Ronald C. Petersen
Gerald H. Peterson
John B. Posey
Gary S. Robinson
Akio Tojo
Donald W. Zipse

*Member Emeritus

Also included is the following nonvoting IEEE-SA Standards Board liaison:

Alan Cookson, *NIST Representative*
Donald R. Volzka, *TAB Representative*

Yvette Ho Sang
IEEE Standards Project Editor

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Information Technology — Portable Operating System Interface (POSIX®) — Test methods for measuring conformance to POSIX —

Part 1: System interfaces

Amendment 1: Realtime Extension (C Language)

Section 1: General

1.1 Scope

This standard defines the test method specifications for measuring conformance of an implementation to POSIX.1b {3}[2]. Assertions are the primary test method specification for measuring conformance to POSIX.1b {3}. Conformance to POSIX.1b {3} requires conformance to IEEE Std 1003.1b-1993 as it has been modified by POSIX.1b {3}. Therefore, the test method specifications for POSIX.1b {3} assume that the test method specifications of IEEE Std 2003.1-1992 {4} apply, except as modified by this standard.

This standard is intended for use by

(1) Developers of POSIX.1b {3} test methods

(2) Implementors of POSIX.1b {3} implementations

(3) Application writers for POSIX.1b {3} conforming implementations

(4) POSIX.1b {3} testing laboratories

(5) Others interested in validating the conformance of a vendor-claimed POSIX.1b {3} implementation

[2] The numbers in braces correspond to those of the references in 1.2.

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

The purpose of this standard is to specify the assertions and related test method specifications for measuring conformance of an implementation to POSIX.1b {3}.

This standard specifies additions and modifications to IEEE Std 2003.1-1992 {4}.

Testing conformance of an implementation to a standard includes testing the claimed capabilities and behavior of the implementation with respect to the conformance requirements of the standard. These test methods are intended to provide a reasonable, practical assurance that the implementation conforms to the standard. Use of these test methods will not guarantee conformance of an implementation to POSIX.1b {3}; that normally would require exhaustive testing, which is impractical for both technical and economic reasons.

IEEE Std 2003.1b-2000 defines a means of measuring conformance to the POSIX.1b {3} technical specifications. Any question related to interpretation of technical specifications arising from the use of this standard is a question of interpretation of POSIX.1b {3}.

1.2 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 part of this International Standard 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 646: 1991, *Information processing—ISO 7-bit coded character set for information interchange* [3].

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

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

{4} IEEE Std 2003.1-1992, *IEEE Standard for Information Technology—Test Methods for Measuring Conformance to POSIX—Part 1: System Interfaces.*

{5} ISO/IEC 13210:1999 (IEEE Std 2003-1997), *Information technology—Requirements and Guidelines for Test Method Specifications and Test Method Implementations for Measuring Conformance to POSIX Standards.*

[3] ISO/IEC documents are available from the ISO Central Secretariat, Case Postal 56, 1 rue de Varembe, CH-1211, Genève 20, Switzerland/Suisse (http://www.iso.ch/). ISO/IEC publications are available in the United States from Global Engineering Documents, 15 Inverness Way East, Englewood, Colorado 80112, USA (http://global.ihs.com/). Electronic copies are available in the United States from the American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA (http://www.ansi.org/).

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

1.3 Conformance

1.3.1 Implementation Conformance

1.3.1.1 Requirements

There are only IEEE Std 2003.1-1992 {4} assertions in this subclause; there are no POSIX.1b {3} assertions.

1.3.1.2 Documentation

There are only IEEE Std 2003.1 {4} assertions in this subclause; there are no POSIX. 1b {3} assertions.

1.3.1.3 Conforming Implementation Options

There are no requirements for conforming implementations in this subclause.

1.3.2 Application Conformance

There are no requirements for conforming implementations in this subclause.

1.3.3 Language -Dependent Services for the C Programming Language

There are only IEEE Std 2003.1-1992 {4} assertions in this subclause; there are no POSIX.1b {3} assertions.

1.3.3.1 Types of Conformance

There are no requirements for conforming implementations in this subclause.

1.3.3.2 C Standard Language-Dependent System Support

There are only IEEE Std 2003.1-1992 {4} assertions in this subclause; there are no POSIX.1b {3} assertions.

1.3.3.3 Common-Usage C Language-Dependent System Support

(IEEE Std 2003.1-1992 {4} DGA01
UNUSED

M_GD_CommonC_diffs (function) =
FOR: *function ()*
IF the implementation does not provide C Standard {2} support **THEN**
TEST: Subclause 8.1 of the POSIX.1b contains the details of all differences between the interface *function ()* provided and that required by the C Standard {2}.
ELSE *NO_OPTION*

GD_CommonC_diffs
FOR: *function* ()
M_GD_CommonC_diffs (function)
Conformance for conformance: PASS, NO_OPTION

1.3.4 Other C Language-Related Specifications

(IEEE Std 2003.1-1992 {4} GA01
UNUSED

M_GA_macro_args (function; header1; header2; header3; header4) =
IF the interface *function* () is defined as a macro **THEN**
SETUP: The headers `<header1>`, `<header2>`, `<header3>`, and `<header 4>` are included.

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

TEST: When the macro *function* () is invoked with the correct argument types (or compatible argument types in the case that C Standard {2} support is provided), the macro evaluates its arguments only once, fully protected by parentheses when necessary, and protects its result value with extra parentheses when necessary.

ELSE *NO_OPTION*

GA_macro_args

FOR: All interfaces except *abort* (), *assert* (), *getc* (), *putc* (), *setjmp* (), *sig-setjmp* (), and *tzset* ().

M_GA_macro_args (function; header1; header 2; header 3; header 4)

Conformance for conformance: PASS, NO_OPTION

M_GA_macro_result_decl (function_type; function; header1; header2; header3; header4) =

IF the interface *function* () is defined as a macro **THEN**

SETUP: The headers `<header1>`, `<header2>`, `<header3>`, and `<header4>` are included.

TEST: When the macro *function* () is invoked with the correct argument types (or compatible argument types in the case that C Standard {2} support is provided), it expands to an expression with the result type *function_type*.

ELSE *NO_OPTION*

GA_macro_result_decl

FOR: All functions implemented as macros

M_GA_macro_result_decl (function; header1; header2; header3; header4)

Conformance for conformance: PASS, NO_ OPTION

1.3.5 Other Language-Related Specifications

There are no requirements for conforming implementations in this clause.

1.4 Test Methods

This clause defines conventions, terminology, testing methodology, and testing control variables used in this
73 standard.

1.4.1 Conventions

The conventions used in specifying the assertions in this standard follow those of ISO/IEC 13210:1999 {5} with the extensions described below.

One convention used in this document is the use of *PCTS_* variables. Because of the IF... Otherwise structure in POSIX.1b {3}, it was necessary to combine the use of {_POSIX_} compile-time symbolic constants that indicate optional facilities in POSIX.1b {3} with the case where the implementor chooses to provide some of the functions specified to be present when the option is defined. For example, an implementation that provides the *sem_init*() function, but does not provide all facilities required for the {_POSIX_SEMAPHORES_} option to be set, would require a PCTS to be configured with the constant *PCTS_sem_init* to be **TRUE**. (Note, that this constant would also need to be **TRUE** if the {_POSIX_SEMAPHORES_} option is set.)

The following construct means that assertion 03, within the corresponding clause of IEEE Std 2003.1-1992 {4}, is not used by this standard:

(IEEE Std 2003.1-1992 {4} 03

UNUSED

Assertions cited by reference assertions are typically named rather than numbered. An attempt has been made to choose meaningful names.

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Because the syntactic conventions used in this standard are those of ISO/IEC 13210:1999 {5}, general assertions and general documentation assertions are rewritten in this document, even if they are also found in IEEE Std 2003.1-1992 {4}.

1.4.1.1 Phrases

The following phrases are commonly used in this standard; they have the indicated meanings.

There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

This means that under the immediately preceding heading there are no new conformance requirements specified in POSIX.1b {3} that need to be tested; hence, there are no assertions. It does not mean that there are requirements for a conforming implementation that were specified in IEEE Std 1003.1b-1993 that must be satisfied. The assertions for those requirements are specified in IEEE Std 2003.1-1992 {4}.

There are no requirements for conforming implementations in this clause.

This means that there are no requirements for a conforming implementation specified in either POSIX.1b {3} or in IEEE Std 1003.1b-1993 under the immediately preceding heading.

1.4.1.2 Macros

This standard uses macros in order to save space and to make it easier to read. Macros are given long meaningful names so that the reader can know what test is covered by the assertion the macro represents. The key to reading a macro is to read and understand its definition. Macro definitions are given by the macro name followed by an equals sign (=). The following is a macro definition for an assertion that specifies the test for the proper C Standard {2} prototype being declared:

M_GA_stdC_proto_decl(func_type; func_name; param1, param2, ...; header1; header2; header3; header 4)=

IF standard THEN

SETUP: The headers `<header1>, <header2>, <header3>,` and `<header 4>` are included.

TEST: The function prototype *func_type func_name(param1, param2, ...)* is declared.

ELSE *NO_OPTION*

When a macro is used to indicate a test for a specific function, the reader should substitute the parameters in the macro call for those in the same position in the macro definition.

1.4.1.3 Cross References

Each interface definition in POSIX.1b {3} contains a Section Cross-References clause, that lists other, related parts of the standard. None of these Section Cross-References clauses contain any requirements for conforming implementations; to conserve printing expense, the corresponding sections are omitted in this standard.

1.4.2 Abbreviations

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

1.4.2.1 C Standard: ISO/IEC 9899:1990, *Programming languages–C* {2}.

1.4.2.2 IRV: The International Reference Version coded character set described in ISO/IEC 646:1991 {1}.

1.4.2.3 POSIX.1b: IEEE Std 1003.1b-1993 {3}.

1.4.2.4 POSIX.1tm: IEEE Std 2003.1-1992 {4}.

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

1.4.2.5 POSIX.tm: ISO/IEC 13210:1999 {5}.

1.4.2.6 POSIX.1b: POSIX Conformance Document, as specified in clause 1.3.1.2 of IEEE Std 1003.1b-1993 as amended by POSIX.1b {3}.

1.4.2.7 IUT: Implementation under test, the software that implements POSIX.1b {3}.

1.4.3 PCTS Variables

The variables that control what assertions to test and indicate which facilities an implementation provides are defined in Table 1-1.

There is at least one PCTS variable for each of the new interfaces defined by POSIX.1b {3}. The PCTS variable can be TRUE depending on one of two conditions: either the defined POSIX.1b {3} option (for example, POSIX_SEMAPHORES) is TRUE, or the implementation supports the interfaces even though it does not necessarily support all of the functionality associated with the implementation option. By combining these two conditions in the definition of the PCTS variable, the expression of conditions under which to test an assertion is made significantly easier.

Some of the interface variables contain the string "_GAP_" or "_RAP_". These strings indicate that the interface variable should be TRUE if there is a way to "get appropriate privilege" and to "release appropriate privilege," respectively.

It is recommended that a conforming test method provide a checklist, or the equivalent, to help users specify the values of the following PCTS symbols.

Table 1-1 – PCTS Symbols and Values

Symbol	Value
PCTS_AIO_CANCELABLE_OPS	**TRUE** if there are I/O operations that can be canceled by calling aio_cancel(), else **FALSE**.
PCTS_AIO_MAX	The lesser of {ARG_MAX}, as obtained from *sysconf*(), and 10 times {_POSIX_ARG_MAX}.
PCTS_APPEND_WRITE_SAME_ORDER	**TRUE** if the implementation does not relax the ordering restriction on asynchronous I/O appends occurring in the same order they were issued, else **FALSE**.
PCTS_CHMOD_SGID	**TRUE** if there are no implementation-defined restrictions that will cause the S_ISGID bit to be ignored in the *mode* argument to a *chmod*() call, else **FALSE**.
PCTS_CHMOD_SUID	**TRUE** if there are no implementation-defined restrictions that will cause the S_ISUID bit to be ignored in the mode *argument to a* chmod() call, else **FALSE**.
PCTS_DELAYTIMER_MAX	The lesser of {DELAYTIMER_MAX}, as obtained from *sysconf*(), and 8 times {POSIX_DELAYTIMER_MAX }.

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Symbol	Value
PCTS_DETECT_AIO_ERROR_AIOCBP	**TRUE** if the implementation can detect that an *aiocbp* argument does not refer to an asynchronous I/O operation whose return status has not yet been retrieved.
PCTS_DETECT_ENOSPC	**TRUE** if the implementation can detect that an out-of-space condition occurs when writing to a device containing the specified file, else **FALSE**.
PCTS_DETECT_EPERM_CLOCK_SETTIME	**TRUE** if the implementation can detect that a process does not have the appropriate privileges to set the specified clock in a call to the *clock_settime* () function, else **FALSE**.
PCTS_DETECT_INVALID_FLAGS_MMAP	**TRUE** if the implementation can detect that the arguments of *addr* or *off* are not multiples of {PAGESIZE} in a call to the *mmap*() function, else **FALSE.**
PCTS_DETECT_LOCKABLE_MEMORY_LIMIT_MLOCK	**TRUE** if the implementation can detect that locking the pages mapped by the specified range would exceed an implementation-defined limit on the amount of memory that a process may lock, else **FALSE**.
PCTS_DETECT_LOCKABLE_MEMORY_LIMIT_MLOCKALL	**TRUE** if the implementation can detect that locking all of the pages currently mapped into the address space of a process would exceed an implementation-defined limit on the amount of memory that a process may lock, else **FALSE**.
PCTS_DETECT_MESSAGE_DATA_CORRUPTION	**TRUE** if the implementation can detect that a data corruption problem with a message in a message queue, else **FALSE.**
PCTS_DETECT_NOT_MULTIPLE_OF_PAGESIZE	**TRUE** if the implementation can detect that the *addr* argument is not a multiple of the page size {PAGESIZE}, else **FALSE**.
PCTS_DETECT_NO_AP	**TRUE** if the implementation can detect that the calling process does not have appropriate privilege to perform the requested operation, else **FALSE**. Used by the *mlock*() and *mlockall*() functions.

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Symbol	Value
PCTS_EINVAL_fchmod	**TRUE** if the implementation can detect that the *fildes* argument refers to a pipe and the implementation disallows execution of *fchmod*() on a pipe, else **FALSE.**
PCTS_EXTEND_ON_ftruncate	**TRUE** if the implementation extends a file to the *length* specified in a call to the *ftruncate*() function if the file previously was smaller that this size, else **FALSE**.
PCTS_GAP_mlock	**TRUE** if a process can get the appropriate privilege to lock process memory with *mlock*(), else **FALSE**.
PCTS_GAP_mlockall	**TRUE** if a process can get the appropriate privilege to lock process memroy with *mlockall*(), else **FALSE**.
PCTS_GAP_MODES_fchmod	**TRUE** if a process can get the appropriate privilege to change the file permission bits of a file using *fchmod*(), else **FALSE**.
PCTS_GAP_mq_open	**TRUE** if a process can get the appropriate privilege to send and receive messages in the message queue specified in *mq_open*(), else **FALSE**.
PCTS_GAP_sem_init	**TRUE** if a process can get the appropriate privilege to initialize a semaphore using *sem-init*(), else **FALSE**.
PCTS_GAP_SGID_fchmod	**TRUE** if a process can get the appropriate privilege, when the effective user ID of the calling process does not match the file owner, to change the S_ISGID file permission bits of the file using *fchmod*(), else **FALSE.**
PCTS_GAP_sigqueue	**TRUE** if a process can get the appropriate privilege to change send a signal to the receiving process using *sigqueue*(), else **FALSE.**
PCTS_GAP_SUID_fchmod	**TRUE** if a process can get the appropriate privilege, when the effective user ID of the calling process does not match the file owner, to change the S_ISUID file permission bits of the file using *fchmod*(), else **FALSE.**
PCTS_GAP_clock_settime	**TRUE** if a process can get the appropriate privilege to set a particular clock using *clock-settime*(), else **FALSE.**

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Symbol	Value
PCTS_GAP_sched_setparam	**TRUE** if a process can get the appropriate privilege to set its own scheduling parameters or those of another using *sched_setparam*(), else **FALSE.**
PCTS_GAP_sched_setscheduler	**TRUE** if a process can get the appropriate privilege to change the scheduling parameters of another process or itself using *sched-setscheduler*(), else **FALSE.**
PCTS_GTI_DEVICE	**TRUE** if the implementation provides device types that support the General Terminal Interface, else **FALSE.**
PCTS_INVALID_SIGNAL	**TRUE** if the implementation has an invalid signal number, else **FALSE.**
PCTS_MAP_FIXED	**TRUE** if the implementation supports the use of the MAP_FIXED mode of memory mapping, else **FALSE.**
PCTS_MAP_PRIVATE	**TRUE** if the implementation supports the facilities indicated by the MAP_PRIVATE flat defined in the `<sys/mman.h>`, else **FALSE.**
PCTS_MORE_SA_SIGINFO_SIGNALS	**TRUE** if the implementation supports the setting of the *si_code* member of the *siginfo_t* structure by means other than calling *kill*(), *raise*(), and *abort*() (if *si_code* is set to SI_USER), *sigqueue*(), or *timer_settime*(), or by completion of an asynchronous I/O request, or by the arrival of a message on an empty message queue, else **FALSE.**
PCTS_MQ_AS_FILE_TYPE	**TRUE** if the implementation supports message queues as file type, else **FALSE.**
PCTS_MQ_OPEN_MAX	The lesser of {MP_OPEN_MAX}, as obtained from *sysconf*IO, and 256.
PCTS_MULTIPLE_OF_PAGESIZE	**TRUE** if the implementation requires that *addr* be a multiple of the page size, {PAGESIZE}, else **FALSE**.
PCTS_NAME_MAX	The lesser of {NAME_MAX}, as obtained from *pathconf*(), and 2048.
PCTS_NO_SYNC_IO_FILE	**TRUE** if the implementation has file types for which synchronized I/O is not supported, else **FALSE.**

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Symbol	Value
PCTS_OPEN_MAX	The lesser of {OPEN_MAX}, as obtained from *sysconf*(), and 256.
PCTS_PATH_MAX	The lesser of {PATH_MAX}, as obtained from *sysconf*(), and 4096.
PCTS_PIPE_BUF	The lesser of {PIPE_BUF}, as obtained from *sysconf*(), and 32767.
PCTS_RAP_mlock	**TRUE** if the implementation supports releasing appropriate privilege for *mlock*(), else **FALSE.**
PCTS_RAP_mlockall	**TRUE** if the implementation supports releasing appropriate privilege for *mlockall*(), else **FALSE.**
PCTS_RAP_mq_open	**TRUE** if the implementation supports releasing appropriate privilege for *mq_open*(), else **FALSE.**
PCTS_RAP_sem_init	**TRUE** if the implementation supports releasing appropriate privileges for *sem_init*(), else **FALSE.**
PCTS_RAP_SGID_chmod	**TRUE** if the implementation supports releasing appropriate privilege for managing the S_ISGID bit in a call to *chmod*(), else **FALSE.**
PCTS_RAP_SGID_fchmod	**TRUE** if the implementation supports releasing appropriate privilege for managing the S_ISUID bit in a call to *fchmod*(), else **FALSE.**
PCTS_RAP_sigqueue	**TRUE** if the implementation supports releasing appropriate privilege for *sigqueue*(), else **FALSE.**
PCTS_RAP_clock_settime	**TRUE** if the implementation supports releasing appropriate privilege for *clock-settime*(), else **FALSE.**
PCTS_RAP_sched_setparam	**TRUE** if the implementation supports releasing appropriate privilege for *sched-setparam*(), else **FALSE.**
PCTS_ROFS	**TRUE** if the implementation supports read-only file systems, else **FALSE.**
PCTS_SEM_EBUSY	**TRUE** if the implementation supports the detection of the [EBUSY] error condition for semaphores, else **FALSE**.

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Symbol	Value
PCTS_SEM_INVALID	**TRUE** if the implementation supports a way to obtain an invalid semaphore, else **FALSE.**
PCTS_SEM_IS_FD	**TRUE** if the implementation supports semaphores as a file type, else **FALSE.**
PCTS_SEM_NSEMS_MAX	The lesser of {SEM_SEMS_MAX}, as obtained from *sysconf*(), and 1024.
PCTS_SHM_AS_FILE_TYPE	**TRUE** if the implementation supports shared memory objects as a distinct file type, else **FALSE.**
PCTS_SIGQUEUE_MAX	The lesser of {SIGQUEUE_MAX}, as obtained from *sysconf*(), and 64.
PCTS_SIGTIMEDWAIT_VALUE	**TRUE** if the implementation can detect when the *sigtimedwait*() function is called with a *timeout* argument specifying a *tv_nsec* value less than zero or greater than or equal to 1000 million, else **FALSE**.
PCTS_TIMER_MAX	The lesser of {TIMER_MAX}, as obtained from *sysconf*(), and 256.
PCTS_UNSUPPORTED_SIGNAL	**TRUE** if the implementation has an unsupported signal number else **FALSE.**
PCTS_aio_cancel	**TRUE** if _POSIX_ASYNCHRONOUS_IO is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_aio_error	**TRUE** if _POSIX_ASYNCHRONOUS_IO is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_aio_fsync	**TRUE** if _POSIX_ASYNCHRONOUS_IO and _POSIX_SYNCHRONIZED_IO are defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_aio_read	**TRUE** if _POSIX_ASYNCHRONOUS_IO is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Symbol	Value
PCTS_aio_return	**TRUE** if _POSIX_ASYNCHRONOUS_IO is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_aio_suspend	**TRUE** if _POSIX_ASYNCHRONOUS_IO is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_aio_write	**TRUE** if _POSIX_ASYNCHRONOUS_IO is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_clock_getres	**TRUE** if POSIX_TIMERS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_clock_gettime	**TRUE** if _POSIX_TIMERS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_clock_settime	**TRUE** if _POSIX_TIMERS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_fchmod	**TRUE** if _POSIX_MAPPED_FILES or _POSIX_SHARED_MEMORY_OBJECTS are defined, or the implementation supports the function as described in POSIX.1b{3}, else **FALSE**.
PCTS_fdatasync	**TRUE** if _POSIX_SYNCHRONIZED_IO is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_fsync	**TRUE** if _POSIX_ASYNCHRONOUS_IO and _POSIX_SYNCHRONIZED_IO are defined or the implementation supports the function as described in POSIX.1b{3}, else **FALSE.**
PCTS_ftruncate	**TRUE** if _POSIX_MAPPED_FILES or _POSIX_SHARED_MEMORY_OBJECTS are defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_get_priority_max	**TRUE** if _POSIX_PRIORITY_SCHEDULING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Symbol	Value
PCTS_get_priority_min	**TRUE** if _POSIX_PRIORITY_SCHEDULING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_lio_listio	**TRUE** if _POSIX_ASYNCHRONOUS_IO is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_mlock	**TRUE** if _POSIX_MEMLOCK_RANGE_ is defined or the implementation supports the function as described in POSIX.1b{3}, else **FALSE**.
PCTS_mlockall	**TRUE** if _POSIX_MEMLOCK is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_mmap	**TRUE** if _POSIX_MAPPED_FILES or _POSIX_SHARED_MEMORY_OBJECTS are defined, or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_mprotect	**TRUE** if _POSIX_MEMORY_PROTECTION is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_mq_close	**TRUE** if _POSIX_MESSAGE_PASSING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_mq_getattr	**TRUE** if _POSIX_MESSAGE_PASSING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_mq_notify	**TRUE** if _POSIX_MESSAGE_PASSING and _POSIX_REALTIME_SIGNALS are defined, or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_mq_open	**TRUE** if _POSIX_MESSAGE_PASSING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Symbol	Value
PCTS_mq_receive	**TRUE** if _POSIX_MESSAGE_PASSING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_mq_send	**TRUE** if _POSIX_MESSAGE_PASSING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_mq_setattr	**TRUE** if _POSIX_MESSAGE_PASSING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_mq_unlink	**TRUE** if _POSIX_MESSAGE_PASSING is defined or the implementation supports the function as described in POSIX.1b {3], else **FALSE.**
PCTS_msync	**TRUE** if _POSIX_MAPPED_FILES and _POSIX_SYNCHRONIZED_IO are defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_msync_storage	**TRUE** if the system under test has secondary storage to which *msync*() can synchronize pages, else **FALSE**.
PCTS_munlock	**TRUE** if _POSIX_MEMLOCK_RANGE is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_munlockall	**TRUE** if _POSIX_MEMLOCK is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_munmap	**TRUE** if _POSIX_MAPPED_FILES or _POSIX_SHARED_MEMORY_OBJECTS are defined, or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_nanosleep	**TRUE** if _POSIX_TIMERS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_read	**TRUE** if the implementation supports the *read*() function (always **TRUE**).

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Symbol	Value
PCTS_sched_get_priority_max	**TRUE** if _POSIX_PRIORITY_SCHEDULING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_sched_get_priority_min	**TRUE** if _POSIX_PRIORITY_SCHEDULING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_sched_getparam	**TRUE** if _POSIX_PRIORITY_SCHEDULING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_sched_getscheduler	**TRUE** if _POSIX_PRIORITY_SCHEDULING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_sched_rr_get_interval	**TRUE** if _POSIX_PRIORITY_SCHEDULING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_sched_setparam	**TRUE** if _POSIX_PRIORITY_SCHEDULING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_sched_setscheduler	**TRUE** if _POSIX_PRIORITY_SCHEDULING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_sched_yield	**TRUE** if _POSIX_PRIORITY_SCHEDULING is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_sem_close	**TRUE** if _POSIX_SEMAPHORES is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_sem_destroy	**TRUE** if _POSIX_SEMAPHORES is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Symbol	Value
PCTS_sem_getvalue	**TRUE** if _POSIX_SEMAPHORES is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_sem_init	**TRUE** if _POSIX_SEMAPHORES is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_sem_open	**TRUE** if _POSIX_SEMAPHORES is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_sem_post	**TRUE** if _POSIX_SEMAPHORES is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_sem_trywait	**TRUE** if _POSIX_SEMAPHORES is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_sem_unlink	**TRUE** if _POSIX_SEMAPHORES is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_sem_wait	**TRUE** if _POSIX_SEMAPHORES is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_shm_open	**TRUE** if _POSIX_SHARED_MEMORY _OBJECTS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_shm_unlink	**TRUE** if _POSIX_SHARED_MEMORY _OBJECTS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_sigqueue	**TRUE** if _POSIX_REALTIME_SIGNALS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_sigtimedwait	**TRUE** if _POSIX_REALTIME_SIGNALS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Symbol	Value
PCTS_sigwaitinfo	**TRUE** if _POSIX_REALTIME_SIGNALS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE.**
PCTS_timer_create	**TRUE** if _POSIX_TIMERS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_timer_delete	**TRUE** if _POSIX_TIMERS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_timer_getoverrun	**TRUE** if _POSIX_TIMERS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_timer_gettime	**TRUE** if _POSIX_TIMERS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_timer_settime	**TRUE** if _POSIX_TIMERS is defined or the implementation supports the function as described in POSIX.1b {3}, else **FALSE**.
PCTS_write	**TRUE** if the implementation supports the *write* () function. (always **TRUE**).

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Documents
Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Section 2: Terminology and General Requirements

2.1 Conventions

This standard uses the following typographic conventions:

(1) The *italic* font is used for

-- Cross references to defined terms within 1.3, 2.2.1, and 2.2.2, symbolic parameters that are generally substituted with real values by the application

-- C language data types and function names (except in function Synopsis clauses)

-- Global external variable names

-- General Assertion and General Documentation Assertion references.

(2) The **bold** font is used with a word in all capital letters, such as **PATH**, to represent an environmental variable. It is also used for the term **"NULL** pointer."

(3) The `constant-width` (Courier) font is used

- For C language data types and function names within function Synopsis clauses
- To illustrate examples of system input or output where exact usage is depicted
- For references to utility names and C language headers.

(4) Symbolic constants returned by many functions as error numbers are represented as

[ERRNO]

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

[LIMIT]

(6) Test method macros are represented as **M_test_method_macro ().**

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

The conventions listed above 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; that is, the comma is used as a decimal sign, and units are from the International System (SI).

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

2.2 Definitions

2.2.1 Terminology

There are no requirements for conforming implementations in this clause.

2.2.2 General Terms

There are no requirements for conforming implementations in this clause.

2.2.2.1 absolute pathname: There are no requirements for conforming implementations in this clause.

2.2.2.2 access mode: There are no requirements for conforming implementations in this clause.

2.2.2.3 address space: There are no requirements for conforming implementations in this clause.

2.2.2.4 appropriate privileges: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.5 arm (a timer): There are no requirements for conforming implementations in this clause.

2.2.2.6 asynchronous I/O completion: There are no requirements for conforming implementations in this clause.

2.2.2.7 asynchronous I/O operation: There are no requirements for conforming implementations in this clause.

2.2.2.8 background process: There are no requirements for conforming implementations in this clause.

2.2.2.9 background process group: There are no requirements for conforming implementations in this clause.

2.2.2.10 block special file: There are no requirements for conforming implementations in this clause.

2.2.2.11 blocked process: There are no requirements for conforming implementations in this clause.

2.2.2.12 character: There are no requirements for conforming implementations in this clause.

2.2.2.13 character special file: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.14 child process: There are no requirements for conforming implementations in this clause.

2.2.2.15 clock: There are no requirements for conforming implementations in this clause.

2.2.2.16 clock tick: There are no requirements for conforming implementations in this clause.

2.2.2.17 controlling process: There are no requirements for conforming implementations in this clause.

2.2.2.18 controlling terminal: There are no requirements for conforming implementations in this clause.

2.2.2.19 current working directory: There are no requirements for conforming implementations in this clause.

2.2.2.20 device: There are no requirements for conforming implementations in this clause.

2.2.2.21 directory: There are no requirements for conforming implementations in this clause.

2.2.2.22 direct I/O: There are no requirements for conforming implementations in this clause.

2.2.2.23 directory entry [link]: There are no requirements for conforming implementations in this clause.

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

2.2.2.24 disarm (a timer): There are no requirements for conforming implementations in this clause.

2.2.2.25 drift rate (of a clock): There are no requirements for conforming implementations in this clause.

2.2.2.26 dot: There are no requirements for conforming implementations in this clause.

2.2.2.27 dot-dot: There are no requirements for conforming implementations in this clause.

2.2.2.28 effective group ID: There are no requirements for conforming implementations in this clause.

2.2.2.29 effective user ID: There are no requirements for conforming implementations in this clause.

2.2.2.30 empty directory: There are no requirements for conforming implementations in this clause.

2.2.2.31 empty string [null string]: There are no requirements for conforming implementations in this clause.

2.2.2.32 Epoch: There are no requirements for conforming implementations in this clause.

2.2.2.33 feature test macro: There are no requirements for conforming implementations in this clause.

2.2.2.34 FIFO special file [FIFO]: There are no requirements for conforming implementations in this clause.

2.2.2.35 file: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.36 file description: There are no requirements for conforming implementations in this clause.

2.2.2.37 file descriptor: There are no requirements for conforming implementations in this clause.

2.2.2.38 file group class: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.39 file mode: There are no requirements for conforming implementations in this clause.

2.2.2.40 filename:
(IEEE Std 2003.1-1992 {4} GA02
UNUSED

M_GA_*portableFilenames*(*function*()) =
TEST: **The interface *function ()*** supports filenames containing any of the characters in the portable filename character set.
TR: Test for filenames containing the following characters:
A B D C E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z
0 1 2 3 4 5 6 7 8 9 . _ -
The last three characters are the period, underscore, and hyphen, respectively.

GA_portableFilenames
FOR: *execl(), execle(), execv(), execve(), exelp(), execvp(), opendir(), chdir(), open(), creat(), link(), existing, link() new, mkdir(), mkfifo(), unlink(), rmdir(), rename(), old, rename(), new, stat(), access(), chmod(), chown(), utime(), pathconf(), fopen(), freopen(), remove(),* `tar` format creating utility, and `cpio` format creating utility.
M_GA_portableFilenames(function())
Conformance for definition: PASS

(IEEE Std 2003.1-1992 {4} GA03
UNUSED

M_GA_upperLowerNames*(function()) =*
TEST: The interface *function*() differentiates between uppercase and lowercase characters in filenames.

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

GA_upperLowerNames

FOR: *execl(), execle(), execv(), execve(), execlp(), execvp(), opendir(), chdir(), open(), creat(), link() existing, link() new, mkdir(), mkfifo(), unlink(), rmdir(), rename() old, rename() new, stat(), access(), chmod(), chown(), utime(), pathconf(), fopen(), freopen(), remove(),* `tar` format creating utility, and `cpio` format creating utility.

M_GA_upperLowerNames*(function())*

Conformance for definitions: PASS

2.2.2.41 file offset: There are no requirements for conforming implementations in this clause.

2.2.2.42 file other class: There are no requirements for conforming implementations in this clause.

2.2.2.43 file owner class: There are no requirements for conforming implementations in this clause.

2.2.2.44 file permission bits: There are no requirements for conforming implementations in this clause.

2.2.2.45 file serial number: There are no requirements for conforming implementations in this clause.

2.2.2.46 file system: There are no requirements for conforming implementations in this clause.

2.2.2.47 first open (of a file): `cpio` format creating utility. There are no requirements for conforming implementations in this clause.

2.2.2.48 foreground process: There are no requirements for conforming implementations in this clause.

2.2.2.49 foreground process group: There are no requirements for conforming implementations in this clause.

2.2.2.50 foreground process group ID: There are no requirements for conforming implementations in this clause.

2.2.2.51 group ID: There are no requirements for conforming implementations in this clause.

2.2.2.52 job control: There are no requirements for conforming implementations in this clause.

2.2.2.53 last close (of a file) There are no requirements for conforming implementations in this clause.

2.2.2.54 link: There are no requirements for conforming implementations in this clause.

2.2.2.55 link count: There are no requirements for conforming implementations in this clause.

2.2.2.56 login: There are no requirements for conforming implementations in this clause.

2.2.2.57 login name: There are no requirements for conforming implementations in this clause.

2.2.2.58 map: There are no requirements for conforming implementations in this clause.

2.2.2.59 memory object: There are no requirements for conforming implementations in this clause.

2.2.2.60 memory-resident: There are no requirements for conforming implementations in this clause.

2.2.2.61 message: There are no requirements for conforming implementations in this clause.

2.2.2.62 message queue: There are no requirements for conforming implementations in this clause.

2.2.2.63 mode: There are no requirements for conforming implementations in this clause.

2.2.2.64 null string: There are no requirements for conforming implementations in this clause.

2.2.2.65 open file: There are no requirements for conforming implementations in this clause.

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

2.2.2.66 open file description: There are no requirements for conforming implementations in this clause.

2.2.2.67 orphaned process group: There are no requirements for conforming implementations in this clause.

2.2.2.68 page: There are no requirements for conforming implementations in this clause.

2.2.2.69 parent directory: There are no requirements for conforming implementations in this clause.

2.2.2.70 parent process: There are no requirements for conforming implementations in this clause.

2.2.2.71 parent process ID: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.72 path prefix: There are no requirements for conforming implementations in this clause.

2.2.2.73 pathname: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.74 pathname component: There are no requirements for conforming implementations in this clause.

2.2.2.75 persistence: There are no requirements for conforming implementations in this clause.

2.2.2.76 pipe: There are no requirements for conforming implementations in this clause.

2.2.2.77 portable filename character set: There are no requirements for conforming implementations in this clause.

2.2.2.78 preallocation: There are no requirements for conforming implementations in this clause.

2.2.2.79 preempted process: There are no requirements for conforming implementations in this clause.

2.2.2.80 priority: There are no requirements for conforming implementations in this clause.

2.2.2.81 priority-based scheduling: There are no requirements for conforming implementations in this clause.

2.2.2.82 privilege: There are no requirements for conforming implementations in this clause.

2.2.2.83 process: There are no requirements for conforming implementations in this clause.

2.2.2.84 process group: There are no requirements for conforming implementations in this clause.

2.2.2.85 process group ID: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.86 process group leader: There are no requirements for conforming implementations in this clause.

2.2.2.87 process group lifetime: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.88 process ID: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.89 process lifetime: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.90 process list: There are no requirements for conforming implementations in this clause.

2.2.2.91 read-only file system: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.92 real group ID: There are no requirements for conforming implementations in this clause.

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

2.2.2.93 real user ID: There are no requirements for conforming implementations in this clause.

2.2.2.94 referenced shared memory object: There are no requirements for conforming implementations in this clause.

2.2.2.95 regions: There are no requirements for conforming implementations in this clause.

2.2.2.96 regular file: There are no requirements for conforming implementations in this clause.

2.2.2.97 relative pathname: There are no requirements for conforming implementations in this clause.

2.2.2.98 resolution (time): There are no requirements for conforming implementations in this clause.

2.2.2.99 root directory: There are no requirements for conforming implementations in this clause.

2.2.2.100 runnable process: There are no requirements for conforming implementations in this clause.

2.2.2.101 running process: There are no requirements for conforming implementations in this clause.

2.2.2.102 saved set-group-ID: There are no requirements for conforming implementations in this clause.

2.2.2.103 saved set-user-ID: There are no requirements for conforming implementations in this clause.

2.2.2.104 scheduling: There are no requirements for conforming implementations in this clause.

2.2.2.105 scheduling policy:

D_1 TEST: The PCD.1b shall define in clause 2.2.2.105 the manner in which each of the scheduling policies may modify the priorities or otherwise affect the ordering of processes at each of the following occurrences

(1) When a process is a running process and it becomes a blocked process

(2) When a process is a running process and it becomes a preempted process

(3) When a process is a blocked process and it becomes a runnable process

(4) When a running process calls a function that can change the priority or scheduling policy of a process

(5) In other scheduling–policy-defined circumstances.

Conformance for definitions: PASS

D_2 TEST: The PCD.1b defines in clause 2.2.2.105 under what other circumstances and in what manner each scheduling policy may modify the priorities or otherwise affect the ordering of processes.

Conformance for definitions: PASS

2.2.2.106 seconds since the Epoch: There are no requirements for conforming implementations in this clause.

2.2.2.107 semaphore: There are no requirements for conforming implementations in this clause.

2.2.2.108 semaphore lock operation:

R_1 FOR: *sem_init()* and *sem_open()*

IF *PCTS_sem_wait* ***THEN***

IF *PCTS_function* **THEN**

SETUP: Create a semaphore using *function* ().

Copyright © 2003 IEEE. All rights reserved.

Documents Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

TEST: When a call to *sem_wait (sem)* completes successfully, the interface returns a value of 0, and the semaphore designated by *sem* is locked by the semaphore lock operation.

TR: When testing for *sem_init*(), perform the test consistent with the flag *PCTS_GAP_sem_init;* that is, generate a *NO_TEST_SUPPORT* test result code if there is no way to get appropriate privilege to call *sem_init*().

NOTE: The assertion is to be tested once for each function specified in the FOR clause. The assertion is to be read by substituting *function*() with the current function specified in the FOR clause. The name of the function also is to be substituted for each occurrence in the construct *PCTS_function.*

ELSE *NO_TEST_SUPPORT*

ELSE *NO_OPTION*

SEE: Assertion sem_wait in 11.2.7.2

2.2.2.109 semaphore unlock operation:

R_2 FOR: *sem_init()* and *sem_open()*

IF *PCTS_sem_post* ***THEN***

IF *PCTS_function* **THEN**

SETUP: Create a semaphore using *function* ().

TEST: A successful call to *sem_post*() unlocks the semaphore referenced by *sem* by performing the semaphore unlock operation on that semaphore, and returns the value zero.

TR: When testing for *sem_init*(), perform the test consistent with the flag *PCTS_GAP_sem_init;* that is, generate a *NO_TEST_SUPPORT* test result code if there is no way to get appropriate privilege to call *sem_init*().

NOTE: The assertion is to be tested once for each function specified in the FOR clause. The assertion is to be read by substituting *function*() with the current function specified in the FOR clause. The name of the function also is to be substituted for each occurrence in the construct *PCTS_function*

ELSE *NO_TEST_SUPPORT*

ELSE *NO_OPTION*

SEE: Assertion sem_post in 11.2.7.2

2.2.2.110 session: There are no requirements for conforming implementations in this clause.

2.2.2.111 session leader: There are no requirements for conforming implementations in this clause.

2.2.2.112 session lifetime: There are no requirements for conforming implementations in this clause.

2.2.2.113 shared memory object: There are no requirements for conforming implementations in this clause.

2.2.2.114 signal: There are no requirements for conforming implementations in this clause.

2.2.2.115 slash: There are no requirements for conforming implementations in this clause.

2.2.2.116 supplementary group ID: There are only IDDD Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.2.2.117 successfully transferred: There are no requirements for conforming implementations in this clause.

2.2.2.118 synchronized I/O completion: There are no requirements for conforming implementations in this clause.

2.2.2.119 synchronized I/O data integrity completion:

GA_syncIODataIntegrityRead

FOR: *read(), aio_read(),* and *lio_listio()*

IF *PCTS_function* and {POSIX_SYNCH_IO} **THEN**

SETUP: Open a file by calling *open*() with O_RSYNC and O_DSYNC set in the *oflag* parameter.

Copyright © 2003 IEEE – All rights reserved

Documents Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

TEST: At the time that the synchronized read operation initiated by calling *function*() occurs, any pending write requests affecting the data to be read are written to the physical medium containing the file prior to reading the data.

TR: Test for regular files.

ELSE *NO_OPTION*

Conformance for definitions: PASS, NO_TEST, NO_OPTION

GA_syncIODataIntegrityWbeforeR

FOR: *write(), aio_write(),* and *lio_liosio()*

IF *PCTS_function* and {POSIX_SYNCH_IO} **THEN**

SETUP: Open a file by calling *open()* with O_DSYNC set in the *oflag* parameter.

TEST: A write operation initiated by calling *function*() either completes by transferring an image of the data to the physical medium containing the file or, if unsuccessful, by diagnosing and returning an indicator of the error.

TR: Test for regular files and, if PCTS_GTI_DEVICE, terminals.

ELSE *NO_OPTION*

Conformance for definitions: PASS, NO_TEST, NO_OPTION

GA_syncIODataIntegrityWrite

FOR: *write*(), *aio_write*(), and *lio_listio*()

IF *PCTS_funtion and* {POSIX_SYNCH_IO} **THEN**

SETUP: Open a file by calling *open*() with O_DSYNC set in the *oflag* parameter.

TEST: A write operation initiated by calling *function*() either completes by transferring an image of the date to the physical medium containing the file or, if unsuccessful, by diagnosing and returning an indicator of the error.

TR: Test for regular files and, if PCTS_GTI_DEVICE, terminals.

ELSE *NO_OPTION*

Conformance for definitions: PASS, NO_TEST, NO_OPTION

2.2.2.120 synchronized I/O file integrity completion:

GA_syncIOFileIntegrityRead

FOR: *read(), aio_read(),* and *lio_listio()*

IF *PCTS_function* and {POSIX_SYNCH_IO} **THEN**

SETUP Open a file by calling *open()* with O_RSYNC and O_SYNC set in the *oflag* parameter.

TEST: At the time that the synchronized read operation initiated by calling *function()* occurs, any pending write requests affecting the data to be read are written to the physical medium containing the file prior to reading the data, and the following file attributes are also written to the physical medium containing the file prior to returning to the calling process:

1. File mode
2. File serial number
3. ID of device containing this file
4. Number of links
5. User ID of the owner of the file
6. Group ID of the group of the file
7. The file size in bytes
8. Time of last access

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

9. Time of last data modification

10. Time of last file status change

TR: Test for regular files.

ELSE *NO_OPTION*

Conformance for definitions: PASS, NO_TEST, NO_OPTION

GA_syncIOFileIntegrityWrite

FOR: write(), aio_write(), *and* lio_listio()

IF *PCTS_function* and {POSIX_SYNCH_IO} **THEN**

TEST: At the time that the synchronized write operation initiated by calling *function*() occurs, the data are written to the physical medium containing the file, and the following file attributes are also written to the physical medium containing the file prior to returning to the calling process:

1. File mode
2. File serial number
3. ID of device containing this file
4. Number of links
5. User ID of the owner of the file
6. Group ID of the group of the file
7. The file size in bytes
8. Time of last access
9. Time of last data modification
10. Time of last file status change

TR: Test for regular files

ELSE *NO_OPTION*

Conformance for definitions: PASS, NO_TEST, NO_OPTION

2.2.2.121 synchronized I/O operation: There are no requirements for conforming implementations in this clause.

2.2.2.122 synchronous I/O operation: There are no requirements for conforming implementations in this clause.

2.2.2.123 system: There are no requirements for conforming implementations in this clause.

2.2.2.124 system crash: There are no requirements for conforming implementations in this clause.

2.2.2.125 system process: There are no requirements for conforming implementations in this clause.

2.2.2.126 system reboot:

D_3 TEST: The PCD.1b defines in clause 2.2.2.126 the implementation-defined sequence of events (called a system reboot) that may result in the loss of transitory data; i.e., data that is not saved in permanent storage.

Conformance for definitions: PASS

2.2.2.127 terminal [terminal device]: There are no requirements for conforming implementations in this clause.

2.2.2.128 timer: There are no requirements for conforming implementations in this clause.

2.2.2.129 timer overrun: There are no requirements for conforming implementations in this clause.

Copyright © 2003 IEEE – All rights reserved

Documents Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

2.2.2.130 user ID: There are no requirements for conforming implementations in this clause.

2.2.2.131 user name: There are no requirements for conforming implementations in this clause.

2.2.2.132 working directory [current working directory]: There are no requirements for conforming implementations in this clause.

2.2.3 Abbreviations

There are no requirements for conforming implementations in this clause.

See clause 1.4.2 for abbreviations related to this standard.

2.3 General Concepts

2.3.1 extended security controls: There are only IEEE Std 2003.1-1992 {4} assertions in this clause; there are no POSIX.1b {3} assertions.

2.3.2 file access permissions:
(IEEE Std 2003.1-1992 {4} GA04
UNUSED

M_GA_AP_overrideFileAccess*(function()) =*
IF the IUT provides a mechanism for creating processes with the appropriate privilege to override a file access control mechanism **THEN**
SETUP: The process has appropriate privileges for file access.
TEST: A call to the interface *function*() that needs read, write, or search access to the *pathname* argument is granted access to the file when access would otherwise be denied.
ELSE *NO_OPTION*

GA_AP_overrideFileAccess
FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execlp(), execvp(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), rename() new, rename() old, rmdir(), stat(), unlink(),* and *utime().*
M_GA_AP_overrideFileAccess*(function())*
Conformance for General Concepts; PASS, NO_TEST, NO_OPTION

(IEEE Std 2003.1-1992 {4} GA05
UNUSED

M_GA_AP_overrideExecAccess*(function()) =*
IF the IUT provides a mechanism for creating processes with the appropriate privilege to override a file access control mechanism **THEN**
SETUP: The process has appropriate privilege for the file access, and execute permission is granted to at least one user of the file.
TEST: A call to the interface *function*() that needs execute permission to the *path* or *file* argument is granted execute access to the file when access would otherwise be denied.
ELSE *NO_OPTION*

GA_AP_overrideExecAccess
FOR: *execl(), execle(), execv(), execve(), execlp(),* and *execvp().*
M_GA_AP_overrideExecAccess*(function())*
Conformance for General Concepts: PASS, NO_TEST, NO_OPTION

(IEEE Std 2003.1-1992 {4} GA06
UNUSED

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

M_GA_AP_classAccess*(function()) =*

IF the IUT provides a mechanism for creating processes with the appropriate privilege to override a file access control mechanism **THEN**

SETUP: The process does not have appropriate privilege to override the file access control mechanism, and the process requires read, write, execute, or search access to the *pathname* or *file* argument of the interface *function*().

TEST: A call to the interface *function*() is granted access to the file when the required access permission bit is set for the class (file owner class, file group class, or file other class) to which the process belongs.

ELSE *NO_OPTION*

GA_AP_classAccess

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execlp(), execvp(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), rename() new, rename() old, rmdir(), stat(), unlink(),* and *utime().*

M_GA_AP_ClassAccess*(function())*

Conformance for General Concepts: PASS, NO_TEST, NO_OPTION

(IEEE Std 2003.1-1992 {4} GA07

UNUSED

M_GA_AdditionalAccessControl*()) =*

IF IUT provides additional file access control mechanisms **THEN**

TEST: Any additional file access control mechanism shall only further restrict the access permissions defined by the file permission bits.

ELSE *NO_OPTION*

GA_AdditionalAccessControl

MP_GA_AdditionalAccessControl()

Conformance for General Concepts: PASS, NO_TEST, NO_OPTION

(IEEE Std 2003.1-1992 {4} GA08

UNUSED

MP_GA_AlternateAccessControl() =

IF IUT provides alternate file access control mechanisms **THEN**

TEST: Any alternate file access control mechanism specifies file permission bits for the file owner class, file group class, and file other class of the file corresponding to the access permissions to be returned by *stat*() and *fstat****().***

ELSE *NO_OPTION*

GA_AlternateAccessControl

M_GA_AlternateAccessControl()

Conformance for General Concepts, PASS, NO_TEST, NO_OPTION

(IEEE Std 2003.1-1992 {4}) GA09

UNUSED

(IEEE Std 2003.1-1992 {4}) GA10

UNUSED

M_GA_AltAccessEnable () =

IF IUT provides alternate file access control mechanisms **THEN**

TEST: The alternate file access control mechanisms can be enabled only by explicit user action, on a per-file basis, by the file owner or a user with the appropriate privilege.

ELSE *NO_OPTION*

GA_AltAccessEnable

M_GA_AltAccessEnable()

Copyright © 2003 IEEE – All rights reserved

Documents Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Conformance for General Concepts: PASS, NO_TEST, NO_OPTION

(IEEE Std 2003.1-1992 {4}) GA11
UNUSED

M_GA_AltAccessDisable() =
IF IUT provides alternate file access control mechanisms **THEN**
TEST: The alternate file access control mechanisms will be disabled for a file after the file permission bits are changed for that file with *chmod*().
ELSE *NO_OPTION*

GA_AltAccessDisable
M_GA_AltAccessDisable()
Conformance for General Concepts: PASS, NO_TEST, NO_OPTION

(IEEE Std 2003.1-1992 {4}) D03
UNUSED

2.3.3 file hierarchy: There are no requirements for conforming implementations in this clause.

2.3.4 filename portability: There are no requirements for conforming implementations in this clause.

2.3.5 file times update:
(IEEE Std 2003.1-1992 {4})R01
UNUSED

(IEEE Std 2003.1-1992 {4}) GA12
UNUSED

***M_GA_StatTimeUpdate** (function()) =*
TEST: The interface *function*(), when called, updates all time-related fields marked for update and does not update any time-related fields not marked for update.

GA_StatTimeUpdate
FOR: *stat()* and *fstat()*.
***M_GA_StatTimeUpdate**(function())*
Conformance for General Concepts: PASS

***M_GA_NoOpenTimeUpdate**(function()) =*
TEST: All fields that are marked for update are updated when the file is no longer open by any process.

GA_NoOpenTimeUpdate
FOR: *close()* and *fclose*().
***M_GA_NoOpenTimeUpdate** (function())*
Conformance for General Concepts: PASS

NOTE: This assertion is missing in 2003.1.

(IEEE Std 2003.1-1992 {4}_GA13
UNUSED

***M_GA_NoROFSTimeUpdate**(function()) =*
TEST: Time-related field updates are not done for files on read-only file systems.

GA_NoROFSTimeUpdate
FOR: *acc(), chmod(), chown(), creat(), link() existing, link() new, mkdir(), mkfifo(), open(), rename() new, rename() old, rmdir(), unlink(),* and *utime()*.
***M_GA_noROFSTimeUpdate**(function())*

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

Conformance for General Concepts: PASS

2.3.6 pathname resolution:

NOTE: In each of the pathname resolution General Assertions below, for the elements *rmdir(), rename()* new, and *unlink(),* the current working directory should be an empty directory in order to preclude the occurrence of avoidable error conditions. Some implementations will consider the attempt to remove the current working directory an error, and will indicate this with the error indication.

(IEEE Std 2003.1-1992 {4}) GA14

UNUSED

M_GA_PRDot*(function()) =*

TEST: A call to the interface *function()* with a *path* or *file* argument where the first filename component is "." and the argument does not begin with a "/". [slash resolves the *path* or *file* argument by locating the second filename component (when specified) in the current working directory.]

GA_PRDot

FOR: *access(), chdir(), chmod(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execlp(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink()* and *utime().*

M_GA_PRDot*(function())*

Conformance for General Concepts: PASS

(IEEE Std 2003.1-1992 {4}_ GA15

UNUSED

M_GA_PRSlash*(function()) =*

TEST: A call to the interface *function(),* with a *path* or *file* argument pointing to the string "/", resolves the *path* or *file* argument to the root directory of the process.

GA_PRSlash

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execp(), execpt(), fopen(), freopen(), open(), opendir(), oink() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(),* and *utime().*

M_GA_PRSlash*(function())*

Conformance for General Concepts: pass

(IEEE Std 2003.1-1992 {4}) GA16

UNUSED

M_GA_PR3Slash*(function()) =*

TEST: A call to the interface *function*(), with a *path* or *file* argument pointing to the string "///", resolves the *path* or *file* argument to the root directory of the process.

GA_PR3Slash

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execlp(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(),* and *utime().*

M_GA_PR3Slash*Ifunction())*

Conformance for General Concepts: PASS

(IEEE Std 2003.1-1992 {4}) GA17

UNUSED

M_GA_PRSlashsPath*(function()) =*

TEST: A call to the interface *function(),* with a *path* or *file* argument pointing to a string that starts with either a single slash ("/") or three or more slashes, resolves the *path* or *file* argument

Copyright © 2003 IEEE – All rights reserved

Documents Document Sharing Hub

Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

by locating the first filename component of the argument in the root directory of the process.

GA_PRSlashsPath

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execlp(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(),* and *utime().*

M_GA_PRSlashsPath*(function())*

Conformance for General Concepts: PASS

(IEEE Std 2003.1-1992 {4}) GA18

UNUSED

M_GA_PRDotDot*(function()) =*

TEST: A call to the interface *function() (),* with a *path* or *file* argument, where the first filename component is "..", the argument does not begin with a "/" (slash), and the current working directory is not the root directory of the process, resolves the *path* or *file* argument by locating the second filename component (when specified) in the parent directory of the current working directory.

GA_PRDotDot

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execle(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(), and utime().*

M_GA_PRDotDot*(function())*

Conformance for General Concepts; PASS

(IEEE Std 2003.1-1992 {4}) GA19

UNUSED

M_GA_PRRelativeSlash*(function()) =*

TEST: A call to the interface *function() (),* with a *path* or *file* argument pointing to the string "F1/", and where "F1" is a directory, resolves the *path* or *file* argument by locating F1 "F1" in the current working directory.

GA_PRRelativeSlash

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execle(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(), and utime().*

M_GA_PRRelativeSlash*(function())*

Conformance for General Concepts: PASS

(IEEE Std 2003.1-1992 {4} GA20

UNUSED

M_GA_PRRelativeSlashSlash*(function()) =*

TEST: A call to the interface *function* () () *with a path or file argument pointing to the string* "F1//" and "F1" is a directory resolves the *path* or *file* argument by locating "F1" in the current working directory.

GA_PRRelativeSlashSlash

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execle(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(), and utime().*

M_GA_PRRelativeSlashSlash*(function())*

Conformance for General Concepts: PASS

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

M_GA_PRRenameRelativeSlashSlash*(function()) =*

TEST: A call to the interface *function() ()*, with a *new* argument pointing to the string "F1//", and where "F1" is an empty directory, resolves the *new* argument by locating "F1" in the current working directory.

GA_PRRenameRelativeSlashSlash

FOR: *rename()new*

M_GA_PRRenameRelativeSlashSlash*(function())*

Conformance for general Concepts: PASS

(IEEE Std 2003.1-1992 {4})GA21

UNUSED

M_GA_PRRelativeCWD*(function()) =*

TEST: A call to the interface *function() ()*, with a *path* or *file* argument pointing to the string "F1/F2", resolves the *path* or *file* argument by locating "F2" in the directory "F1" in the current working directory.

GA_PRRelativeCWD

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execle(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(), and utime().*

M_GA_PRRelativeCWD*(function())*

Conformance for General Concepts: PASS

(IEEE Std 2003.1-1992 {4}_ GA22

UNUSED

M_GA_PRRelativeDotCWD*(function()) =*

TEST: A call to the interface *function() ()*, with a *path* or *file* argument pointing to the string "F1/./F2", resolves the *path* or *file* argument by locating "F2" in the directory "F1" in the current working directory.

GA_PRRelativeDotCWD

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execle(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(), and utime().*

M_GA_PRRelativeDotCWD*(function())*

Conformance for General Concepts: PASS

(IEEE Std 2003.1-1992 {4} GA23

UNUSED

M_GA_PRRelativeDotDotCWD*(function()) =*

TEST: A call to the interface *function() ()*, with a *path* or *file* argument pointing to the string "F1/../F1/F2", resolves the *path* or *file* argument by locating "F2" in the directory "F1" in the current working directory.

GA_PRRelativeDotDotCWD

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execle(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(), and utime().*

M_GA_PRRelativeDotDotCWD*(function())*

Conformance for General Concepts: PASS

(IEEE Std 2003.1-1992 {4}) GA24

UNUSED

M_GA_PRRelativeSlashSlashCWD*(function()) =*

Copyright © 2003 IEEE – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003

TEST: A call to the interface *function() ()*, with a *path* or *file* argument pointing to the string "F1//F2", resolves the *path* or *file* argument by locating "F2" in the directory "F1" in the current working directory.

GA_PRRelativeSlashSlashCWD

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execle(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(), and utime().*

M_GA_PRRelativeSlashSlashCWD*(function())*

Conformance for General Concepts: PASS

(IEEE Std 2003.1-1992 {4}) GA25

UNUSED

M_GA_PPRnoTrunc*(function()) =*

IF {POSIX_NO_TRUNC} is not supported in the specified directory **THEN**

TEST: A call to the interface *function() ()*, with a *path* or *file* argument that has a pathname component of more than {NAME_MAX} bytes in a directory for which {_POSIX_NO_TRUNC} is not supported, resolves the pathname component by truncating it to {NAME_MAX} bytes.

ELSE *NO_OPTION*

GA_PRNoTrunc

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execle(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(), and utime().*

M_GA_PRNoTrunc*(function())*

Conformance for General Concepts: PASS, NO_OPTION

M_GA_PRnoTruncError*(function()) =*

IF {POSIX_NO_TRUNC is supported in the specified directory **THEN**

TEST: A call to the interface *function() ()*, with a *path* or *file* argument that has a pathname component of more than {NAME_MAX} bytes in a directory for which {POSIX_NO_TRUNC} is supported, generates an [ENAMETOOLONG] error.

ELSE *NO_OPTION*

GA_PRNoTruncError

FOR: *access(), chdir(), chmod(), chown(), creat(), execl(), execle(), execv(), execve(), execle(), execvp(), fopen(), freopen(), open(), opendir(), link() existing, link() new, mkdir(), mkfifo(), pathconf(), remove(), rename() new, rename() old, rmdir(), stat(), unlink(), and utime().*

M_GA_PRNoTruncError*(function())*

Conformance for General Concepts: PASS, NO_OPTION

2.4 Error Numbers

(IEEE Std 2003.1-1992 P{4}) 02

UNUSED

2 **SETUP:** Include the header `<errno.h>`.

TEST: The error numbers[E2BIG], [EACCESS], [EAGAIN[, [EBADF], EBADMSG], EBUSY],]ECANCELED], [ECHILD], [EDEADLK], [EDOM], [EEXIST], [EFAULT[, [EFBIG], [EINPROGRESS], [EINTR], [EINTR], [EINVAL], [EIO], [EISDIR], [EMFILE], [EMLINK], [EMSGSIZE], [ENAMETOOLONG], [ENFILE], [ENODEV], [ENOENT], [ENOEXEC], [EN0LOCK], [ENOMEM], [ENOSPC], [ENOSYS], [ENOTDIR], [ENOTEMPTY], [ENOTTY], [ENXIO], [EPERM], [EPIPE], [ERANGE], [EROFS], [ESPIPE], [ESRCH], and [EXDEV] are defined, are nonzero, are distinct from each other, and can be represented in *errno*.

Copyright © 2003 IEEE. All rights reserved.
Documents Document Sharing Hub
Full document details: ISO_IEC 14515-1_2000_Amd 1_2003