

INTERNATIONAL  
STANDARD

ISO/IEC/  
IEEE  
9945

First edition  
2009-09-15

Corrigendum 2  
2017-03

**Information technology — Portable  
Operating System Interface (POSIX®)  
Base Specifications, Issue 7**

**TECHNICAL CORRIGENDUM 2**

*Technologies de l'information — Spécifications de base de l'interface  
pour la portabilité des systèmes (POSIX®), Issue 7*

*RECTIFICATIF TECHNIQUE 2*

IECNORM.COM : Click to view the full PDF of ISO/IEC/IEEE 9945:2009/Cor.2:2017



Reference number  
ISO/IEC/IEEE 9945:2009/Cor.2:2017(E)



© IEEE 2016



**COPYRIGHT PROTECTED DOCUMENT**

© IEEE 2016

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO or IEEE at the address below or ISO's member body in the country of the requester.

ISO copyright office  
Ch. de Blandonnet 8 • CP 401  
CH-1214 Vernier, Geneva, Switzerland  
Tel. +41 22 749 01 11  
Fax +41 22 749 09 47  
copyright@iso.org  
www.iso.org

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

stds.ipr@ieee.org  
www.ieee.org

## Foreword

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.

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.

The main task of ISO/IEC JTC 1 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 called to the possibility that implementation of this standard may require the 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. ISO/IEEE is not responsible for identifying essential patents or patent claims for which a license may be required, for conducting inquiries into the legal validity or scope of patents or patent claims or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance or a Patent Statement and Licensing Declaration Form, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from ISO or the IEEE Standards Association.

ISO/IEC/IEEE 9945 was prepared by The Open Group (as The Open Group Technical Standard Base Specifications, Issue 7) and the Portable Applications Standards Committee of the Computer Society of the IEEE (as IEEE Std 1003.1™-2008). It was adopted by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*, in parallel with its approval by the ISO/IEC national bodies, under the “fast-track procedure” defined in the Partner Standards Development Organization cooperation agreement between ISO and IEEE. IEEE is responsible for the maintenance of this document with participation and input from ISO/IEC national bodies.

IECNORM.COM : Click to view the full PDF of ISO/IEC/IEEE 9945:2009/Cor 2:2017

**IEEE Std 1003.1™-2008/Cor 2-2016**

(Corrigendum to  
IEEE Std 1003.1-2008)

The Open Group Technical Standard  
Base Specifications, Issue 7

# **IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)**

## **Base Specifications, Issue 7— Technical Corrigendum 2**

Sponsor

**Portable Applications Standards Committee  
of the  
IEEE Computer Society**

and  
**The Open Group**

Approved 30 June 2016

**IEEE-SA Standards Board**

Approved 20 June 2016

**The Open Group**

IECNORM.COM. Click to view the full PDF of ISO/IEC/IEEE 9945:2009/Cor 2:2017

**Abstract:** Problems discovered since the approval of IEEE Std 1003.1-2008 and IEEE Std 1003.1-2008/Cor 1-2013 are addressed.

**Keywords:** API, application program interface, argument, asynchronous, basic regular expression, batch job, batch system, BRE, built-in utility, byte, child, command language interpreter, CPU, ERE, extended regular expression, FIFO, file access control mechanism, IEEE 1003.1™, input/output, I/O, job control, network, portable operating system interface, POSIX®

---

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

Copyright © 2016 by The Institute of Electrical and Electronics Engineers, Inc.  
All rights reserved. Published 15 August 2016. Printed in the United States of America.

Published 15 August 2016 by The Open Group.

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

POSIX is a registered trademark of IEEE.

The Open Group  
Apex Plaza, Forbury Road, Reading, Berkshire RG1 1AX, U.K.

PDF: ISBN 978-1-5044-2148-5 STD1005

*IEEE prohibits discrimination, harassment, and bullying.  
For more information, visit <http://www.ieee.org/web/aboutus/whatis/policies/p9-26.html>.*

*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.*

IEEE Std 1003.1™-2008/Cor 2-2016  
IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)  
Base Specifications, Issue 7—Technical Corrigendum 2

**The Open Group** is a global consortium that enables the achievement of business objectives through IT standards. With more than 500 member organizations, The Open Group has a diverse membership that spans all sectors of the IT community—customers, systems and solutions suppliers, tool vendors, integrators, and consultants, as well as academics and researchers—to:

- Capture, understand, and address current and emerging requirements, establish policies, and share best practices
- Facilitate interoperability, develop consensus, and evolve and integrate specifications and open source technologies
- Offer a comprehensive set of services to enhance the operational efficiency of consortia
- Operate the industry's premier certification service

Further information on The Open Group is available at [www.opengroup.org](http://www.opengroup.org).

The Open Group publishes a wide range of technical documentation, most of which is focused on development of Open Group Standards and Guides, but which also includes white papers, technical studies, certification and testing documentation, and business titles. Full details and a catalog are available at [www.opengroup.org/bookstore](http://www.opengroup.org/bookstore).

Readers should note that updates—in the form of Corrigenda—may apply to any publication. This information is published at [www.opengroup.org/corrigenda](http://www.opengroup.org/corrigenda).

## Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading “Important Notice” or “Important Notices and Disclaimers Concerning IEEE Standards Documents.”

### Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

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

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to: results; and workmanlike effort. IEEE standards documents are supplied “AS IS” and “WITH ALL FAULTS.”

Use of an IEEE standard is wholly voluntary. The existence of an IEEE standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

### Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE should be considered the approved IEEE standard.

IEEE Std 1003.1™-2008/Cor 2-2016  
 IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)  
 Base Specifications, Issue 7—Technical Corrigendum 2

## Official statements

A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, or be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

## Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide consulting information or advice pertaining to IEEE Standards documents. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive 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 comments or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

Secretary, IEEE-SA Standards Board  
 445 Hoes Lane  
 Piscataway, NJ 08854 USA

## Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

## Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

## Photocopies

Subject to payment of the appropriate fee, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, 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.

## Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every ten years. When a document is more than ten years old and has not undergone a revision process, 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 order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE-SA Website at <http://standards.ieee.org> or contact IEEE at the address listed previously. For more information about the IEEE SA or IEEE's standards development process, visit the IEEE-SA Website at <http://standards.ieee.org/develop/index.html>.

## Errata

Errata, if any, for all IEEE standards can be accessed on the IEEE-SA Website at the following URL: <http://standards.ieee.org/findstds/errata/index.html>. Users are encouraged to check this URL for errata periodically.

## Patents

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 by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE-SA Website at <http://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

## Participants

IEEE Std 1003.1™-2008/Cor 2-2016 was prepared by the Austin Group, sponsored by the Portable Applications Standards Committee of the IEEE Computer Society, The Open Group, and ISO/IEC JTC 1/SC22.

### The Austin Group

At the time this draft standard was completed, the Austin Group had the following membership:

**Andrew Josey, Chair**

**Donald W. Cragun**, *Organizational Representative, IEEE PASC*  
**Nicholas M. Stoughton**, *Organizational Representative, ISO/IEC JTC 1/SC22*  
**Roger Faulkner**, *Organizational Representative, The Open Group*

**Cathy Fox, Technical Editor**

### Austin Group Technical Reviewers

Bogdan Barbu	Felix Janda	Martin Řehák
Eric Blake	Aurelio Jargas	Torvald Riegel
Harvey Block	Andrew Josey	Xavier Roche
Mark S. Brown	John Kearney	Askar Safin
Milan Cermak	Michael Kerrisk	Anton Salikhmetov
Stephane Chazelas	Rob King	Jörg Schilling
Alexander Cherepanov	Andi Kleen	Ed Schouten
Mark Clancy	Bruce Korb	Konrad Schwarz
Geoff Clare	Kaz Kylheku	Jens Schweikhardt
David Clissold	Antoine Leca	Mike Scudder
Alan Coopersmith	Wojtek Lerch	Martin Sebor
Donald W. Cragun	Sven Mascheck	Nicholas M. Stoughton
Matthew Dempsky	Margot Hackett Miller	Keith Thompson
Dan Douglas	Joseph S. Myers	Jilles Tjoelker
Lawrence D.K.B. Dwyer	Szabolcs Nagy	William Toth
Paul Eggert	Alexander Nasonov	Daniel Trebbien
Steve Emmerson	Jonathan Nieder	Miloslav Trmac
David Egan Evans	Danny Niu	Fred J. Tydeman
Roger Faulkner	Gian Ntzik	Ted Unangst
Richard Felker	Steffen Nurpmeso	Brian Utterback
Thorsten Glaser	Carlos O'Donnell	Stijn van Drongelen
Glenn D. Golden	Vladimir Támaro Patiño	Jonathan Wakely
Jim Grisanzio	Peter Petrov	Nathan Weeks
Philip Guenther	William Pitcock	Florian Weimer
Richard Hansen	Jim Pryor	David A. Wheeler
Mark Harris	James C. Pugsley	Michael Wilson
Cyril Hrubis	Yury Pukhalsky	André Zepezauer
Jarmo Jaakkola	Steve Rago	Mark Ziegast
	Chet Ramey	

## Austin Group Working Group Members

Eitan Adler	Philip Guenther	Jim Pryor
Iwan Aucamp	Joseph M. Gwinn	James C. Pugsley
Bogdan Barbu	Bruno Haible	Yury Pukhalsky
David Bartley	Richard Hansen	Steve Rago
Steve Bartolomei	Mark Harris	Chet Ramey
Fabrice Bauzac	Barry E. Hedquist	Martin Řehák
Guido Berhoerster	David Holland	Tom Ridge
Eric Blake	Tom Honermann	Torvald Riegel
Harvey Block	Cyril Hrubis	Xavier Roche
Hans Boehm	Jarmo Jaakkola	Askar Safin
Bill Bogstad	Felix Janda	Anton Salikhmetov
Pádraig Brady	Aurelio Jargas	Jörg Schilling
Davide Brini	Ross Johnson	Ed Schouten
Andries E. Brouwer	Andrew Josey	Konrad Schwarz
Mark S. Brown	John Kearney	Ingo Schwarze
David Butenhof	Dan Kegel	Jens Schweikhardt
Milan Cermak	Michael Kerrisk	Mike Scudder
Stephane Chazelas	Rob King	Martin Sebor
Alexander Cherepanov	Tomas Klacko	Glen Seeds
Mark Clancy	Andi Kleen	Jeffrey Sheinberg
Geoff Clare	Bruce Korb	Thor Lancelot Simon
David Clissold	David Korn	Keld Simonsen
Alan Coopersmith	Kaz Kylheku	Ranjit Singh
Donald W. Cragun	Rob Landley	Paul Smith
Martijn Dekker	Antoine Leca	Steven Stewart-Gallus
Matthew Dempsky	Vincent Lefèvre	Nicholas M. Stoughton
Thomas E. Dickey	Sean Leonard	Alfred M. Szmidt
Casper Dik	Wojtek Lerch	Marcel Telka
Dan Douglas	Yonggang Luo	Donn Terry
Niall Douglas	Scott Lurndal	Keith Thompson
Ulrich Drepper	Roger Marquis	Jilles Tjoelker
Lawrence D.K.B. Dwyer	Sven Mascheck	William Toth
Paul Eggert	Per Mildner	Daniel Trebbien
Daniel Eischen	Margot Hackett Miller	Miloslav Trmac
Robert Elz	Joseph S. Myers	Fred J. Tydeman
Steve Emmerson	Szabolcs Nagy	Ted Unangst
Laszlo Ersek	Alexander Nasonov	Brian Utterback
Bruce Evans	Jonathan Nieder	Stijn van Drongelen
David Egan Evans	Danny Niu	Christopher Vance
Roger Faulkner	Gian Ntzik	Jonathan Wakely
Richard Felker	Steffen Nurpmeso	Nathan Weeks
Jeffrey K. Fellin	Carlos O'Donnell	Florian Weimer
Hal Finkel	Isabella Parakiss	David A. Wheeler
Glenn Fowler	Vladimir Támaria Patiño	Mats D. Wichmann
Cathy Fox	Phil Pennock	Michael Wilson
Mike Frysinger	Andres Perera	Garrett Wollman
Andrea Giugliano	Peter Petrov	Jörg Wunsch
Thorsten Glaser	William Pitcock	James Youngman
Glenn D. Golden	Wayne Pollock	André Zepezauer
Jim Grisanzio		Mark Ziegast

IEEE Std 1003.1™-2008/Cor 2-2016  
 IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)  
 Base Specifications, Issue 7—Technical Corrigendum 2

## The Open Group

When The Open Group approved the Base Specifications, Issue 7, Technical Corrigendum 2 on 20 June 2016, the membership of The Open Group Base Working Group was as follows:

**Andrew Josey, Chair**  
**Roger Faulkner, Austin Group Liaison**

**Cathy Fox, Technical Editor**

## Base Working Group Members

Eric Blake  
 Geoff Clare  
 David Clissold

Donald W. Cragun  
 Roger Faulkner  
 Darrin Johnson

Andrew Josey  
 Martin Řehák  
 S. R. Srinivasan

## IEEE

At the time this standard was submitted to the IEEE-SA Standards Board for approval, the Portable Applications Standards Committee had the following membership:

**Joseph M. Gwinn, Chair**  
**Andrew Josey, Functional Chair (Interpretations)**  
**Donald W. Cragun, Shell and Utilities Working Group Chair**  
**Barry Hedquist, Test Methods Working Group Chair**  
**Craig Meyers, Distributed Services Working Group Chair**  
**Stephen Walli, US TAG Institutional Representative**  
**Roger Martin, Ex-officio Emeritus**  
**Nicholas M. Stoughton, Secretary**

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Johann Amsenga  
 Juan Carreon  
 Keith Chow  
 Donald W. Cragun  
 Sourav Dutta  
 Andrew Fieldsend  
 David Fuschi

Randall Groves  
 Joseph M. Gwinn  
 Barry E. Hedquist  
 Werner Hoelzl  
 Noriyuki Ikeuchi  
 Andrew Josey  
 Piotr Karocki

Kenneth Lang  
 Vincent Lefèvre  
 Peter Petrov  
 Stephen Schwarm  
 Walter Struppler  
 Mark-Rene Uchida  
 Oren Yuen

IEEE Std 1003.1™-2008/Cor 2-2016  
IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)  
Base Specifications, Issue 7—Technical Corrigendum 2

When the IEEE-SA Standards Board approved this standard on 30 June 2016, it had the following membership:

**Jean-Philippe Faure, Chair**  
**Ted Burse, Vice Chair**  
**John D. Kulick, Past Chair**  
**Konstantinos Karachalios, Secretary**

Chuck Adams  
Masayuki Ariyoshi  
Stephen Dukes  
Jianbin Fan  
J. Travis Griffith  
Ronald W. Hotchkiss

Gary Hoffman  
Michael Janezic  
Joseph L. Koepfinger\*  
Hung Ling  
Kevin Lu  
Annette D. Reilly  
Gary Robinson

Mehmet Ulema  
Yingli Wen  
Howard Wolfman  
Don Wright  
Yu Yuan  
Daidi Zhong

\*Member Emeritus

IECNORM.COM : Click to view the full PDF of ISO/IEC/IEEE 9945:2009/Cor 2:2017

IEEE Std 1003.1™-2008/Cor 2-2016  
IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)  
Base Specifications, Issue 7—Technical Corrigendum 2

## Introduction

This introduction is not part of IEEE Std 1003.1-2008/Cor 2, IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)/Base Specifications, Issue 7—Technical Corrigendum 2.

This Technical Corrigendum addresses issues raised in defect reports and interpretation requests submitted up to 24 August 2015 that meet all of the following criteria:

- a) They are in the scope of the approved standard.
- b) They contain no new APIs (functions/utilities); however, they may add enumeration symbols, non-function #defines, and reserve additional namespaces.
- c) They address contradictions between different parts of the standard, or add consistency between the standard and overriding standards, or address security-related problems.

## Contents

1. Changes to the Front Matter .....	2
2. Changes to Base Definitions.....	3
3. Changes to System Interfaces.....	46
4. Changes to Shell and Utilities .....	198
5. Changes to Rationale .....	283

IECNORM.COM : Click to view the full PDF of ISO/IEC/IEEE 9945:2009/Cor.2:2017

IEEE Std 1003.1™-2008/Cor 2-2016  
IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)  
Base Specifications, Issue 7—Technical Corrigendum 2

# IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)

## Base Specifications, Issue 7— Technical Corrigendum 2

**IMPORTANT NOTICE:** This standard is not intended to ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. Implementers of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

This IEEE document is made available for use subject to important notices and legal disclaimers. These notices and disclaimers appear in all publications containing this document and may be found under the heading “Important Notice” or “Important Notices and Disclaimers Concerning IEEE Documents.” They can also be obtained on request from IEEE or viewed at <http://standards.ieee.org/IPR/disclaimers.html>.

NOTE—The editing instructions contained in this corrigendum define how to merge the material contained therein into the existing base standard to form the comprehensive standard.

1 **1. Changes to the Front Matter**

2 This section contains the set of changes to the text of the front matter.

3 *[Note to reviewers: References to defect reports are provided to aid reviewers.]*

4 **Change Number: Front Matter/TC2/D4/0001** [885]

5 On Page: xlv Line: NA Section: Referenced Documents  
6 (2013 edition Page: xlvii Line: NA)

7 In the line below the **Source Documents** header, change from:

8 base documents for POSIX.1-2008

9 to:

10 base documents for POSIX.1-2001

11 *Rationale:* Austin Group Defect Report(s) applied: 885. See <http://austingroupbugs.net/view.php?id=885>.

12 **Change Number: Front Matter/TC2/D4/0002** [726]

13 On Page: xxxviii Line: NA Section: Acknowledgements

14 Remove the period from the first bullet list item.

15 *Rationale:* Austin Group Defect Report(s) applied: 726. See <http://austingroupbugs.net/view.php?id=726>.  
16 Editorial change.

17

## 18 2. Changes to Base Definitions

19 This section contains the set of changes to the text of the Base Definitions.  
 20 [Note to reviewers: References to defect reports are provided to aid reviewers.]

21 **Change Number: XBD/TC2/D4/0001** [591]

22 On Page: 9 Line: 266 Section: 1.7.1 Codes

23 Change from:

```
24 [OH]#include <sys/types.h>[ /OH]
25 #include <grp.h>
26 struct group *getgrnam(const char *name);
```

27 The OH margin legend indicates that the marked header is not required on XSI-conformant systems.

28 to:

```
29 [OH]#include <sys/stat.h>[ /OH]
30 #include <fcntl.h>
31 int open(const char *path, int oflag, ...);
```

32 The OH margin legend indicates that the optional header defines constants that will be needed if the  
 33 function is called with certain flag arguments; thus it may be required for some of the functionality  
 34 described, but is not needed otherwise.

35 *Rationale:* Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

36 **Change Number: XBD/TC2/D4/0002** [810]

37 On Page: 16 Line: 493 Section: 2.1.1 Requirements

38 At the end of section 2.1.1, add a small-font note:

39 **Note:** If the documented method of setting up a conforming environment includes the need to set one or  
 40 more environment variables, then the values of those environment variables cannot include any <space>  
 41 characters, since the *confstr()* function must be able to return them in a <space>-separated list of  
 42 variable=value pairs. See [xref to XSH *confstr()*].

43 *Rationale:* Austin Group Defect Report(s) applied: 810. See <http://austingroupbugs.net/view.php?id=810>.

44 **Change Number: XBD/TC2/D4/0003** [637]

45 On Page: 17 Line: 561 Section: 2.1.3 POSIX Conformance

46 Change from:

```
47     _POSIX_TIMERS
48     -- Symbolic constants defined with a value greater than zero:
```

50       \_POSIX\_JOB\_CONTROL  
51       \_POSIX\_SAVED\_IDS  
  
52       to:  
  
53       \_POSIX\_TIMERS  
54       \_POSIX2\_C\_BIND  
55  
56       -- Symbolic constants defined with a value greater than zero:  
57       \_POSIX\_JOB\_CONTROL  
58       \_POSIX\_REGEXP  
59       \_POSIX\_SAVED\_IDS  
60       \_POSIX\_SHELL

61       *Rationale:* Austin Group Defect Report(s) applied: 637. See <http://austingroupbugs.net/view.php?id=637>.

62       **Change Number: XBD/TC2/D4/0004** [937]

63       On Page: 35 Line: 1195 Section: 3.17 Application  
64       (2013 edition Page: 35 Line: 1199)

65       After:

66       A computer program that performs some desired function.

67       add a new paragraph:

68       When the User Portability Utilities option is supported, requirements placed on applications relating to the  
69       use of standard utilities shall also apply to the actions of a user who is entering shell command language  
70       statements into an interactive shell.

71       *Rationale:* Austin Group Defect Report(s) applied: 937. See <http://austingroupbugs.net/view.php?id=937>.

72       **Change Number: XBD/TC2/D4/0005** [516]

73       On Page: 37 Line: 1229 Section: 3.27 Async-Signal-Safe Function

74       Change the definition from:

75       A function that may be invoked, without restriction, from signal-catching functions. No function is async-  
76       signal-safe unless explicitly described as such.

77       to:

78       A function that can be called, without restriction, from signal-catching functions. Note that, although there  
79       is no restriction on the calls themselves, for certain functions there are restrictions on subsequent behavior  
80       after the function is called from a signal-catching function. No function is async-signal-safe unless  
81       explicitly described as such.

82       Note: Async-signal-safety is defined in detail in [xref to XSH 2.4.3].

83       *Rationale:* Austin Group Defect Report(s) applied: 516. See <http://austingroupbugs.net/view.php?id=516>.  
84       The restrictions on using longjmp() and siglongjmp() are more restrictive than they need to be on POSIX

85 systems. The loosened restrictions presented here do not break existing implementations and make it easier  
86 for application writers to create portable applications.

87 **Change Number: XBD/TC2/D4/0006** [653]

88 On Page: 39 Line: 1274 Section: 3.40 Basename

89 Change from:

90 The final, or only, filename in a pathname.

91 to:

92 For pathnames containing at least one filename: the final, or only, filename in the pathname. For pathnames  
93 consisting only of <slash> characters: either "/" or "//" if the pathname consists of exactly two <slash>  
94 characters, and "/" otherwise.

95 *Rationale:* Austin Group Defect Report(s) applied: 653. See <http://austingroupbugs.net/view.php?id=653>.

96 **Change Number: XBD/TC2/D4/0007** [834]

97 On Page: 60 Line: 1775 Section: 3.168 File Mode

98 Change from:

99 An object containing the file mode bits and file type of a file.

100 to:

101 An object containing the file mode bits and some information about the file type of a file.

102 *Rationale:* Austin Group Defect Report(s) applied: 834. See <http://austingroupbugs.net/view.php?id=834>.

103 **Change Number: XBD/TC2/D4/0008** [511]

104 On Page: 63 Line: 1855 Section: 3.188 Group ID

105 Add a sentence:

106 The value `(gid_t)-1` shall not be a valid group ID, but does have a defined use in some interfaces defined in  
107 this standard.

108 *Rationale:* Austin Group Defect Report(s) applied: 511. See <http://austingroupbugs.net/view.php?id=511>.

109 **Change Number: XBD/TC2/D4/0009** [584]

110 On Page: 63 Line: 1859 Section: 3.189 Group Name

111 Change from:

112 <hyphen>

113 to:

114 <hyphen-minus>

115 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

116 **Change Number: XBD/TC2/D4/0010** [690]

117 On Page: 66 Line: 1926 Section: 3 Definitions  
118 (2013 edition Page: 66 Line: 1935)

119 After section 3.208 ("Link Count"), insert a new section **3.209 Live Process** whose contents are copied  
120 from section 3.289 ("Process") page 80 lines 2243-2250 (2013 edition section 3.291 page 80 lines 2269-  
121 2275):

122 An address space with one or more threads executing within that address space, and the required system  
123 resources for those threads.

124 <small>**Note:** Many of the system resources defined by POSIX.1-2008 are shared among all of the threads  
125 within a process. These include the process ID, the parent process ID, process group ID, session  
126 membership, real, effective, and saved set-user-ID, real, effective, and saved set-group-ID, supplementary  
127 group IDs, current working directory, root directory, file mode creation mask, and file descriptors.</small>

128 *Rationale:* Austin Group Defect Report(s) applied: 690. See <http://austingroupbugs.net/view.php?id=690>.

129 **Change Number: XBD/TC2/D4/0011** [625]

130 On Page: 69 Line: 2000 Section: 3 Definitions

131 Add the following definitions in alphabetical order:

132 **Multi-threaded library**

133 A library containing object files that were produced by compiling with *c99* using the flags output by  
134 *getconf POSIX\_V7\_THREADS\_CFLAGS*, or by compiling using a non-standard utility with equivalent  
135 flags, and which makes use of interfaces that are only made available by *c99* when the **-l pthread** option is  
136 used or makes use of *SIGEV\_THREAD* notifications.

137 **Multi-threaded process**

138 A process that contains more than one thread.

139 **Multi-threaded program**

140 A program whose executable file was produced by compiling with *c99* using the flags output by *getconf*  
141 *POSIX\_V7\_THREADS\_CFLAGS*, and linking with *c99* using the flags output by *getconf*  
142 *POSIX\_V7\_THREADS\_LDFLAGS* and the **-l pthread** option, or by compiling and linking using a non-  
143 standard utility with equivalent flags. Execution of a multi-threaded program initially creates a single-  
144 threaded process; the process can create additional threads using *pthread\_create()* or *SIGEV\_THREAD*

145 notifications.

146 **Single-threaded process**

147 A process that contains a single thread.

148 **Single-threaded program**

149 A program whose executable file was produced by compiling with *c99* without using the flags output by  
 150 *getconf POSIX\_V7\_THREADS\_CFLAGS* and linking with *c99* using neither the flags output by *getconf*  
 151 *POSIX\_V7\_THREADS\_LDFLAGS* nor the **-l pthread** option, or by compiling and linking using a non-  
 152 standard utility with equivalent flags. Execution of a single-threaded program creates a single-threaded  
 153 process; if the process attempts to create additional threads using *pthread\_create()* or *SIGEV\_THREAD*  
 154 notifications, the behavior is undefined. If the process uses *dlopen()* to load a multi-threaded library, the  
 155 behavior is undefined.

156 *Rationale*: Austin Group Defect Report(s) applied: 625. See <http://austingroupbugs.net/view.php?id=625>.  
 157 The standard does not adequately address the differences between single-threaded and multi-threaded  
 158 processes and programs.

159 **Change Number: XBD/TC2/D4/0012 [584]**

160 On Page: 77 Line: 2193 Section: 3. Definitions

161 After Section 3.275, add a new section:

162 **3.2xx Portable Filename**

163 A filename consisting only of characters from the portable filename character set.

164 <small>Note: Applications should avoid using filenames that have the <hyphen-minus> character as the  
 165 first character since this may cause problems when filenames are passed as command line  
 166 arguments.</small>

167 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.  
 168 This is a layered change on XBD/TC1/D5/0010 [291]. The change is to replace <hyphen> with <hyphen-  
 169 minus>.

170 **Change Number: XBD/TC2/D4/0013 [584]**

171 On Page: 77 Line: 2199 Section: 3.276 Portable Filename Character Set

172 Change from:

173 <hyphen>

174 to:

175 <hyphen-minus>

176 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

177 **Change Number: XBD/TC2/D4/0014** [690]

178 On Page: 80 Line: 2244-2250 Section: 3.289 Process  
179 (2013 edition Page: 80 Line: 2269-2275 Section: 3.291)

180 Replace the definition with:

181 A live process (see Section 3.xxx) or a zombie process (see Section 3.xxx). The lifetime of a process is  
182 described in Section 3.xxx.

183 On Page: 81 Line: 2270-2279 Section: 3.295 Process Lifetime  
184 (2013 edition Page: 81 Line: 2295-2304 Section: 3.297)

185 Change from:

186 The period of time that begins when a process is created and ends when its process ID is returned to the  
187 system. After a process is created by *fork()*, *posix\_spawn()*, or *posix\_spawnp()*, it is considered active. At  
188 least one thread of control and address space exist until it terminates. It then enters an inactive state where  
189 certain resources may be returned to the system, although some resources, such as the process ID, are still  
190 in use. When another process executes a *wait()*, *waitid()*, or *waitpid()* function for an inactive process, the  
191 remaining resources are returned to the system. The last resource to be returned to the system is the process  
192 ID. At this time, the lifetime of the process ends.

193  
194 <small>**Note:** The *fork()*, *posix\_spawn()*, *posix\_spawnp()*, *wait()*, *waitid()*, and *waitpid()* functions are  
195 defined in detail in the System Interfaces volume of POSIX.1-2008.</small>

196 to:

197 The period of time that begins when a process is created and ends when its process ID is returned to the  
198 system.

199  
200 See also *Live Process* in Section 3.xxx, *Process Termination* in Section 3.xxx, and *Zombie Process* in  
201 Section 3.xxx.

202  
203 <small>**Note:** Process creation is defined in detail in the descriptions of the *fork()*, *posix\_spawn()*, and  
204 *posix\_spawnp()* functions in the System Interfaces volume of POSIX.1-2008.</small>

205 On Page: 81 Line: 2293-2294 Section: 3.297 Process Termination  
206 (2013 edition Page: 81 Line: 2318-2319 Section: 3.299)

207 Change from:

208 <small>**Note:** The *\_exit()*, *\_Exit()*, *abort()*, and *exit()* functions are defined in detail in the System  
209 Interfaces volume of POSIX.1-2008.</small>

210 to:

211 <small>**Note:** The consequences of process termination can be found in the description of the *\_Exit()*  
212 function in the System Interfaces volume of POSIX.1-2008. The *\_exit()*, *\_Exit()*, *abort()*, and *exit()*  
213 functions are defined in detail in the System Interfaces volume of POSIX.1-2008.</small>

214 *Rationale:* Austin Group Defect Report(s) applied: 690. See <http://austingroupbugs.net/view.php?id=690>.

215 **Change Number: XBD/TC2/D4/0015** [896]

216 On Page: 90 Line: 2485 Section: 3.351 Source Code  
217 (2013 edition Page: 90 Line: 2510 Section: 3.353 Source Code)

218 Change from:

219 ... *ed, lex, ...*

220 to:

221 ... *ed, ex, lex, ...*

222 On Page: 90 Line: 2487 Section: 3.351 Source Code  
223 (2013 edition Page: 90 Line: 2512 Section: 3.353 Source Code)

224 Change from:

225 ... *ed, lex, ...*

226 to:

227 ... *ed, ex, lex, ...*

228 *Rationale:* Austin Group Defect Report(s) applied: 896. See <http://austingroupbugs.net/view.php?id=896>.

229 **Change Number: XBD/TC2/D4/0016** [511]

230 On Page: 102 Line: 2782 Section: 3.428 User ID

231 Add a sentence:

232 The value **(uid\_t)-1** shall not be a valid user ID, but does have a defined use in some interfaces defined in  
233 this standard.

234 *Rationale:* Austin Group Defect Report(s) applied: 511. See <http://austingroupbugs.net/view.php?id=511>.

235 **Change Number: XBD/TC2/D4/0017** [584]

236 On Page: 102 Line: 2786 Section: 3.429 User Name

237 Change from:

238 <hyphen>

239 to:

240 <hyphen-minus>

241 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

242

243 **Change Number: XBD/TC2/D4/0018** [690]

244 On Page: 105 Line: 2845-2846 Section: 3.444 Zombie Process  
 245 (2013 edition Page: 105 Line: 2871-2872 Section: 3.446)

246 Change from:

247 A process that has terminated and that is deleted when its exit status has been reported to another process  
 248 which is waiting for that process to terminate.

249 to:

250 The remains of a live process (see Section 3.xxx) after it terminates (see Section 3.xxx) and before its status  
 251 information (see XSH Section 2.13) is consumed by its parent process.

252 *Rationale:* Austin Group Defect Report(s) applied: 690. See <http://austingroupbugs.net/view.php?id=690>.

253 **Change Number: XBD/TC2/D4/0019** [934]

254 On Page: 107 Line: 2858 Section: 4 General Concepts  
 255 (2013 edition Page: 107 Line: 2884)

256 Add a new section (and renumber subsequent sections as needed):

#### 257 **4.2 Default Initialization**

258 Default initialization causes an object to be initialized according to these rules:

- 259 • If it has pointer type, it is initialized to a null pointer.
- 260 • If it has arithmetic type, it is initialized to (positive or unsigned) zero.
- 261 • If it is an aggregate, every member is initialized (recursively) according to these rules.
- 262 • If it is a union, the first named member is initialized (recursively) according to these rules.

263 For an object of aggregate type with an explicit initializer, the initialization shall occur in initializer list  
 264 order, each initializer provided for a particular subobject overriding any previously listed initializer for the  
 265 same subobject; all subobjects that are not initialized explicitly shall be initialized implicitly according to  
 266 the rules for default initialization.

267 Objects with static storage duration but no explicit initializer shall be initialized implicitly according to the  
 268 rules for default initialization.

269 An explicit initializer of { 0 } works to perform explicit default initialization for any object of scalar or  
 270 aggregate type, and for any storage duration.

271 <small>Note: The C standard does not require a compiler to set any field alignment padding bits in a  
 272 structure or array definition to a particular value. Because of this, a structure initialized using { 0 } might  
 273 not *memcmp()* as equal to the same structure initialized using *memset()* to zero. For consistent results,  
 274 portable applications comparing structures should test each field individually.

275 Note: If an implementation treats the all-zero bit pattern of a pointer object as a null pointer, and the all-  
 276 zero bit pattern of a floating point object as equivalent to positive 0, then *memset()* to zero and *calloc()*  
 277 have the same effects as default initialization for all named members of a structure. [MX]Implementations

278 that define `__STDC_IEC_559__` guarantee that the all-zero bit pattern of a floating point object represents  
 279 0.0.[/MX]</small>

280 *Rationale*: Austin Group Defect Report(s) applied: 934. See <http://austingroupbugs.net/view.php?id=934>.

281 **Change Number: XBD/TC2/D4/0020** [584]

282 On Page: 109 Line: 2914 Section: 4.7 Filename Portability

283 Change from:

284 <hyphen>

285 to:

286 <hyphen-minus>

287 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

288 **Change Number: XBD/TC2/D4/0021** [626]

289 On Page: 109 Line: 2917 Section: 4.8 File Times Update

290 Insert a new paragraph at the start of the section:

291 Many operations have requirements to update file timestamps. These requirements do not apply to streams  
 292 that have no underlying file description (for example, memory streams created by `open_memstream()` have  
 293 no underlying file description).

294 *Rationale*: Austin Group Defect Report(s) applied: 626. See <http://austingroupbugs.net/view.php?id=626>.

295 **Change Number: XBD/TC2/D4/0022** [863]

296 On Page: 110 Line: 2980 Section: 4.11 Memory Synchronization  
 297 (2013 edition Page: 110 Line: 3004)

298 Change from:

299 The `pthread_once()` function shall synchronize memory for the first call in each thread for a given  
 300 `pthread_once_t` object.

301 to:

302 The `pthread_once()` function shall synchronize memory for the first call in each thread for a given  
 303 `pthread_once_t` object. If the `init_routine` called by `pthread_once()` is a cancellation point and is canceled,  
 304 a call to `pthread_once()` for the same `pthread_once_t` object made from a cancellation cleanup handler  
 305 shall also synchronize memory.

306 *Rationale*: Austin Group Defect Report(s) applied: 863. See <http://austingroupbugs.net/view.php?id=863>.

307

308   **Change Number: XBD/TC2/D4/0023** [541,649,825]

309   On Page: 112 Line: 3026 Section: 4.12 Pathname Resolution

310   Change from:

311   In all other cases, the system shall prefix the remaining pathname, if any, with the contents of the symbolic link. If the combined length exceeds {PATH\_MAX}, and the implementation considers this to be an error, *errno* shall be set to [ENAMETOOLONG] and an error indication shall be returned.

314   to:

315   In all other cases, the system shall prefix the remaining pathname, if any, with the contents of the symbolic link, except that if the contents of the symbolic link is the empty string, then either pathname resolution shall fail with functions reporting an [ENOENT] error and utilities writing an equivalent diagnostic message, or the pathname of the directory containing the symbolic link shall be used in place of the contents of the symbolic link. If the contents of the symbolic link consist solely of <slash> characters, then all leading <slash> characters of the remaining pathname shall be omitted from the resulting combined pathname, leaving only the leading <slash> characters from the symbolic link contents. In the cases where prefixing occurs, if the combined length exceeds {PATH\_MAX}, and the implementation considers this to be an error, pathname resolution shall fail with functions reporting an [ENAMETOOLONG] error and utilities writing an equivalent diagnostic message.

325   *Rationale:* Austin Group Defect Report(s) applied: 541,649,825. See  
<http://austingroupbugs.net/view.php?id=541>.  
 326   See <http://austingroupbugs.net/view.php?id=649>.  
 327   See <http://austingroupbugs.net/view.php?id=825>.  
 328   The previous text was not historic practice on implementations that support //hostname as a special case in pathname resolution. For implementations that do not treat // as special, the change in handling of all-slash symbolic links has no effect on pathname resolution.

332   **Change Number: XBD/TC2/D4/0024** [825]

333   On Page: 112 Line: 3032 Section: 4.12 Pathname Resolution  
 334   (2013 edition Page: 112 Line: 3056 Section: 4.12)

335   Change from:

336   If the system detects a loop in the pathname resolution process, it shall set *errno* to [ELOOP] and return an error indication.

338   to:

339   If the system detects a loop in the pathname resolution process, pathname resolution shall fail with functions reporting an [ELOOP] error and utilities writing an equivalent diagnostic message.

341   *Rationale:* Austin Group Defect Report(s) applied: 825. See <http://austingroupbugs.net/view.php?id=825>.

342

343 **Change Number: XBD/TC2/D4/0025** [543]

344 On Page: 116 Line: 3205 Section: 4.19 Treatment of Error Conditions for Mathematical Functions

345 Add a new paragraph before the last paragraph:

346 On implementations that support the IEC 60559 Floating-Point option, whether or when functions in the  
 347 <**math.h**> header raise an undeserved underflow floating-point exception is unspecified. Otherwise, as  
 348 implied by [xref to XSH *feraiseexcept()*] the <**math.h**> functions do not raise spurious floating-point  
 349 exceptions (detectable by the user), other than the inexact floating-point exception.

350 *Rationale*: Austin Group Defect Report(s) applied: 543. See <http://austingroupbugs.net/view.php?id=543>.

351 **Change Number: XBD/TC2/D4/0026** [584]

352 On Page: 123 Line: 3411 Section: 5 File Format Notation

353 Change from:

354 indicates either a <plus-sign> or minus-sign),

355 to:

356 indicates either a <plus-sign> or <hyphen-minus>),

357 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

358 **Change Number: XBD/TC2/D4/0027** [584,967]

359 On Page: 125 Line: 3454 Section: 6.1 Portable Character Set  
 360 (2013 edition Page: 125 Line: 3479)

361 Change from:

362 The first eight entries in [xref to Table 6-1] are defined in the ISO/IEC 6429:1992 standard and the rest of  
 363 the characters are defined in the ISO/IEC 10646-1:2000 standard.

364 to:

365 The first eight entries in [xref to Table 6-1] and all characters in [xref to Table 6-2] are defined in the  
 366 ISO/IEC 6429:1992 standard. The rest of the characters in [xref to Table 6-1] are defined in the ISO/IEC  
 367 10646-1:2000 standard.

368 On Page: 125 Line: 3458 Section: 6.1 Portable Character Set  
 369 (2013 edition Page: 125 Line: 3483)

370 Change from:

371 **Symbolic Name**

372 to:

373 **Symbolic Name(s)**

374 On Page: 125 Line: 3460-3466 Section: 6.1 Portable Character Set  
 375 (2013 edition Page: 125 Line: 3485-3491)

376 Change from:

<alert>	<U0007>	BELL (BEL)
<backspace>	<U0008>	BACKSPACE (BS)
<tab>	<U0009>	CHARACTER TABULATION (HT)
<carriage-return>	<U000D>	CARRIAGE RETURN (CR)
<newline>	<U000A>	LINE FEED (LF)
<vertical-tab>	<U000B>	LINE TABULATION (VT)
<form-feed>	<U000C>	FORM FEED (FF)

377

378 to (note the description column changes to match the Unicode spec, and the change in line order):

<alert>, <BEL>	<U0007>	BELL
<backspace>, <BS>	<U0008>	BACKSPACE
<tab>, <HT>	<U0009>	CHARACTER TABULATION
<newline>, <LF>	<U000A>	LINE FEED (LF)
<vertical-tab>, <VT>	<U000B>	LINE TABULATION
<form-feed>, <FF>	<U000C>	FORM FEED (FF)
<carriage-return>, <CR>	<U000D>	CARRIAGE RETURN (CR)

379

380 On Page: 125 Line: 3480-3485 Section: 6.1 Portable Character Set  
 381 (2013 edition Page: 125 Line: 3505-3510)

382 Change from:

IEEE Std 1003.1™-2008/Cor 2-2016  
IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)  
Base Specifications, Issue 7—Technical Corrigendum 2

<hyphen-minus>	-	<U002D>	HYPHEN-MINUS
<hyphen>	-	<U002D>	HYPHEN-MINUS
<full-stop>	.	<U002E>	FULL STOP
<period>	.	<U002E>	FULL STOP
<slash>	/	<U002F>	SOLIDUS
<solidus>	/	<U002F>	SOLIDUS

383

384 to (note that the glyph used for <hyphen-minus>, <hyphen> should be the same one that is used in the  
385 construct "(-)" in the rest of the standard):

<hyphen-minus>, <hyphen>	-	<U002D>	HYPHEN-MINUS
<full-stop>, <period>	.	<U002E>	FULL STOP
<slash>, <solidus>	/	<U002F>	SOLIDUS

386 On Page: 126 Line: 3531-3532 Section: 6.1 Portable Character Set  
387 (2013 edition Page: 126 Line: 3556-3557)

388 Change from:

<backslash>	\	<U005C>	REVERSE SOLIDUS
<reverse-solidus>	\	<U005C>	REVERSE SOLIDUS

389

390 to:

<backslash>, <reverse-solidus>	\	<U005C>	REVERSE SOLIDUS
--------------------------------	---	---------	-----------------

391

392 On Page: 126 Line: 3534-3537 Section: 6.1 Portable Character Set  
393 (2013 edition Page: 126 Line: 3559-3562)

394 Change from:

<circumflex-accent>	^	<U005E>	CIRCUMFLEX ACCENT
---------------------	---	---------	-------------------

<circumflex>	^	<U005E>	CIRCUMFLEX ACCENT
<low-line>	—	<U005F>	LOW LINE
<underscore>	—	<U005F>	LOW LINE

395

396 to:

<circumflex-accent>, <circumflex>	^	<U005E>	CIRCUMFLEX ACCENT
<low-line>, <underscore>	—	<U005F>	LOW LINE

397

398 On Page: 127 Line: 3566-3567 Section: 6.1 Portable Character Set  
 399 (2013 edition Page: 127 Line: 3591-3592)

400 Change from:

<left-brace>	{	<U007B>	LEFT CURLY BRACKET
<left-curly-bracket>	{	<U007B>	LEFT CURLY BRACKET

401

402 to:

<left-brace>, <left-curly-bracket>	{	<U007B>	LEFT CURLY BRACKET
------------------------------------	---	---------	--------------------

403

404 On Page: 127 Line: 3569-3570 Section: 6.1 Portable Character Set  
 405 (2013 edition Page: 127 Line: 3594-3595)

406 Change from:

<right-brace>	}	<U007D>	RIGHT CURLY BRACKET
<right-curly-bracket>	}	<U007D>	RIGHT CURLY BRACKET

407

408 to:

IEEE Std 1003.1™-2008/Cor 2-2016  
 IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)  
 Base Specifications, Issue 7—Technical Corrigendum 2

<right-brace>, <right-curly-bracket>	}	<U007D>	RIGHT CURLY BRACKET
--------------------------------------	---	---------	---------------------

409

410 On Page: 127 Line: 3576-3578 Section: 6.1 Portable Character Set  
 411 (2013 edition Page: 127 Line: 3601-3603)

412 Delete the sentence:

413 The table contains more than one symbolic character name for characters whose traditional name differs  
 414 from the chosen name.

415 *Rationale:* Austin Group Defect Report(s) applied: 584,967. See  
 416 <http://austingroupbugs.net/view.php?id=584>.  
 417 See <http://austingroupbugs.net/view.php?id=967>.

418 **Change Number: XBD/TC2/D4/0028** [745]

419 On Page: 128 Line: 3589 Section: 6.1 Portable Character Set

420 Add a sentence as a new bullet item:

421 • The encoded values associated with <period>, <slash>, <newline> and <carriage-return> shall be  
 422 invariant across all locales supported by the implementation.

423 *Rationale:* Austin Group Defect Report(s) applied: 745. See <http://austingroupbugs.net/view.php?id=745>.  
 424 This is a layered change on XBD/TC1/D5/0019 [291].

425 **Change Number: XBD/TC2/D4/0029** [663,967]

426 On Page: 128 Line: 3596 Section: 6.2 Character Encoding  
 427 (2013 edition Page: 128 Line: 3623)

428 Change from:

429 The POSIX locale contains the characters in [xref to Table 6-1], which have the properties listed in [xref to  
 430 7.3.1]. In other locales, the presence, meaning, and representation of any additional characters are locale-  
 431 specific.

432 to:

433 The POSIX locale shall contain 256 single-byte characters including the characters in [xref to Table 6-1]  
 434 and [xref to Table 6-2], which have the properties listed in [xref to 7.3.1]. It is unspecified whether  
 435 characters not listed in those two tables are classified as **punct** or **cntrl**, or neither. Other locales shall  
 436 contain the characters in [xref to Table 6-1] and may contain any or all of the control characters identified  
 437 in [xref to Table 6-2]; the presence, meaning, and representation of any additional characters are locale-  
 438 specific.

439 *Rationale:* Austin Group Defect Report(s) applied: 663,967. See  
 440 <http://austingroupbugs.net/view.php?id=663>.  
 441 See <http://austingroupbugs.net/view.php?id=967>.

442 This is a layered change on XBD/TC1/D5/0020 [216].  
 443 The intention was always that the POSIX locale should have an 8-bit-clean single-byte encoding. The  
 444 omission of an explicit statement to that effect was an oversight.

445 **Change Number: XBD/TC2/D4/0030 [745]**

446 On Page: 128 Line: 3619 Section: 6.2 Character Encoding

447 Add a sentence:

448 Likewise, the byte values used to encode <period>, <slash>, <newline> and <carriage-return> shall not  
 449 occur as part of any other character in any locale.

450 *Rationale:* Austin Group Defect Report(s) applied: 745. See <http://austingroupbugs.net/view.php?id=745>.  
 451 This is a layered change on XBD/TC1/D5/0021 [291].

452 **Change Number: XBD/TC2/D4/0031 [967]**

453 On Page: 129 Line: 3655 Section: 6.4 Character Set Description File  
 454 (2013 edition Page: 129 Line: 3683)

455 Change from:

456 Each symbolic name specified in [xref to Table 6-1] shall be included in the file and shall be mapped to a  
 457 unique coding value, except as noted below. The glyphs represented by the C character constants '{', '}',  
 458 '\_', '-', '/', '\', '.', and '^' have more than one symbolic name; all symbolic names for each such  
 459 glyph shall be included, each with identical encoding. If some or all of the control characters identified in  
 460 [xref to Table 6-2] are supported by the implementation, the symbolic names and their corresponding  
 461 encoding values shall be included in the file. The encoding values shall each be represented in a single  
 462 byte. Some of the encodings associated with the symbolic names in [xref to Table 6-2] may be the same as  
 463 characters found in [xref to Table 6-1]; both names shall be provided for each encoding.

464 to:

465 Each symbolic name specified in [xref to Table 6-1] shall be included in the file. Each character in [xref to  
 466 Table 6-1] (each row in the table) shall be mapped to a unique coding value. For each character in [xref to  
 467 Table 6-2] that exists in the character set described by the file, the character's symbolic name(s) from [xref  
 468 to Table 6-2] and the character's single-byte encoding value shall be included in the file.

469 On Page: 130 Line: 3664-3670 Section: 6.4 Character Set Description File  
 470 (2013 edition Page: 130 Line: 3692-3698)

471 Change from:

472 **Table 6-2 Control Character Set**

473 <ACK> <DC2> <ENQ> <FS> <IS4> <SOH>  
 <BEL> <DC3> <EOT> <GS> <LF> <STX>

<BS> <DC4> <ESC> <HT> <NAK> <SUB>  
 <CAN> <DEL> <ETB> <IS1> <RS> <SYN>  
 <CR> <DLE> <ETX> <IS2> <SI> <US>  
 <DC1> <EM> <FF> <IS3> <SO> <VT>

474 to:

475 **Table 6-2** Non-Portable Control Characters

476

Symbolic Name(s)	UCS	Description
<SOH>	<U0001>	START OF HEADING
<STX>	<U0002>	START OF TEXT
<ETX>	<U0003>	END OF TEXT
<EOT>	<U0004>	END OF TRANSMISSION
<ENQ>	<U0005>	ENQUIRY
<ACK>	<U0006>	ACKNOWLEDGE
<SO>	<U000E>	SHIFT OUT
<SI>	<U000F>	SHIFT IN
<DLE>	<U0010>	DATA LINK ESCAPE
<DC1>	<U0011>	DEVICE CONTROL ONE
<DC2>	<U0012>	DEVICE CONTROL TWO
<DC3>	<U0013>	DEVICE CONTROL THREE
<DC4>	<U0014>	DEVICE CONTROL FOUR
<NAK>	<U0015>	NEGATIVE ACKNOWLEDGE
<SYN>	<U0016>	SYNCHRONOUS IDLE
<ETB>	<U0017>	END OF TRANSMISSION BLOCK

<CAN>	<U0018>	CANCEL
<EM>	<U0019>	END OF MEDIUM
<SUB>	<U001A>	SUBSTITUTE
<ESC>	<U001B>	ESCAPE
<IS4>, <FS>	<U001C>	INFORMATION SEPARATOR FOUR
<IS3>, <GS>	<U001D>	INFORMATION SEPARATOR THREE
<IS2>, <RS>	<U001E>	INFORMATION SEPARATOR TWO
<IS1>, <US>	<U001F>	INFORMATION SEPARATOR ONE
<DEL>	<U007F>	DELETE

477

478 *Rationale:* Austin Group Defect Report(s) applied: 967. See <http://austingroupbugs.net/view.php?id=967>.

479 **Change Number: XBD/TC2/D4/0032** [796]

480 On Page: 136 Line: 3846 Section: 7.2 POSIX Locale  
 481 (2013 edition Page: 136 Line: 3882)

482 After:

483 Conforming systems shall provide a POSIX locale, also known as the C locale.

484 add:

485 In POSIX.1 the requirements for the POSIX locale are more extensive than the requirements for the C  
 486 locale as specified in the C standard. However, in a conforming POSIX implementation, the POSIX locale  
 487 and the C locale are identical.

488 *Rationale:* Austin Group Defect Report(s) applied: 796. See <http://austingroupbugs.net/view.php?id=796>.

489 **Change Number: XBD/TC2/D4/0033** [663]

490 On Page: 136 Line: 3849 Section: 7.2 POSIX locale  
 491 (2013 edition Page: 136 Line: 3885)

492 Delete:

493 The tables in Section 7.3 describe the characteristics and behavior of the POSIX locale for data consisting  
 494 entirely of characters from the portable character set and the control character set. For other characters, the  
 495 behavior is unspecified.

IEEE Std 1003.1™-2008/Cor 2-2016  
IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)  
Base Specifications, Issue 7—Technical Corrigendum 2

496 On Page: 139 Line: 3996 Section: 7.3.1 LC\_CTYPE  
497 (2013 edition Page: 139 Line: 4032)

498 Change from:

499 In the POSIX locale, the 26 uppercase letters shall be included:

500 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

501 to:

502 In the POSIX locale, only:

503 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

504 shall be included.

505 On Page: 139 Line: 4003 Section: 7.3.1 LC\_CTYPE  
506 (2013 edition Page: 139 Line: 4039)

507 Change from:

508 In the POSIX locale, the 26 lowercase letters shall be included:

509 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

510 to:

511 In the POSIX locale, only:

512 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

513 shall be included.

514 On Page: 139 Line: 4009 Section: 7.3.1 LC\_CTYPE  
515 (2013 edition Page: 140 Line: 4045)

516 Change from:

517 In the POSIX locale, all characters in the classes **upper** and **lower** shall be included.

518 to:

519 In the POSIX locale, only characters in the classes **upper** and **lower** shall be included.

520 On Page: 141 Line: 4091 Section: 7.3.1 LC\_CTYPE  
521 (2013 edition Page: 141 Line: 4127)

522 Change from:

523 In the POSIX locale, at a minimum, the 26 lowercase characters:

524 to:

525 In the POSIX locale, the 26 lowercase characters:

526 On Page: 142 Line: 4105 Section: 7.3.1 LC\_CTYPE  
527 (2013 edition Page: 142 Line: 4141)

528 Change from:

529 In the POSIX locale, at a minimum, the 26 uppercase characters:

530 to:

531 In the POSIX locale, the 26 uppercase characters:

532 *Rationale*: Austin Group Defect Report(s) applied: 663. See <http://austingroupbugs.net/view.php?id=663>.

533 **Change Number: XBD/TC2/D4/0034** [663]

534 On Page: 143 Line: 4142 Section: 7.3.1.1 LC\_CTYPE Category in the POSIX Locale  
535 (2013 edition Page: 143 Line: 4178)

536 Change from:

537 The character classifications for the POSIX locale follow; the code listing depicts the *localedef* input, and  
538 the table represents the same information, sorted by character.

539 to:

540 The minimum character classifications for the POSIX locale follow; the code listing depicts the *localedef*  
541 input, and the table represents the same information, sorted by character. Implementations may add  
542 additional characters to the **cntrl** and **punct** classifications but shall not make any other additions.

543 On Page: 143 Line: 4145 Section: 7.3.1.1 LC\_CTYPE Category in the POSIX Locale  
544 (2013 edition Page: 143 Line: 4181)

545 Change from:

```
546 # The following is the POSIX locale LC_CTYPE.  
547 # "alpha" is by default "upper" and "lower"  
548 # "alnum" is by definition "alpha" and "digit"  
549 # "print" is by default "alnum", "punct", and the <space>  
550 # "graph" is by default "alnum" and "punct"
```

551 to:

```
552 # The following is the minimum POSIX locale LC_CTYPE.  
553 # "alpha" is by definition "upper" and "lower"  
554 # "alnum" is by definition "alpha" and "digit"  
555 # "print" is by definition "alnum", "punct", and the <space>  
556 # "graph" is by definition "alnum" and "punct"
```

557 *Rationale*: Austin Group Defect Report(s) applied: 663. See <http://austingroupbugs.net/view.php?id=663>.

558 **Change Number: XBD/TC2/D4/0035** [584]

559 On Page: 143 Line: 4173 Section: 7.3.1.1 LC\_CTYPE Category in the POSIX Locale

560 Change from:

561 <hyphen>

562 to:

563 <hyphen-minus>

564 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

565 **Change Number: XBD/TC2/D4/0036** [584]

566 On Page: 144 Line: 4243 Section: 7.3.1.1 LC\_CTYPE Category in the POSIX Locale

567 Change from:

568 <hyphen>

569 to:

570 <hyphen-minus>

571 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

572 **Change Number: XBD/TC2/D4/0037** [938]

573 On Page: 146 Line: 4330 Section: 7.3.2 LC\_COLLATE  
 574 (2013 edition Page: 147 Line: 4360)

575 Change from:

576 (*sort, uniq*, and so on)

577 to:

578 (*ls, sort*, and so on)

579 On Page: 147 Line: 4357 Section: 7.3.2 LC\_COLLATE  
 580 (2013 edition Page: 147 Line: 4393)

581 Add a new paragraph and two small-font notes after the numbered list:

582 All implementation-provided locales (either preinstalled or provided as locale definitions which can be  
 583 installed later) should define a collation sequence that has a total ordering of all characters unless the locale  
 584 name has an '@' modifier indicating that it has a special collation sequence (for example, @icase could  
 585 indicate that each upper- and lower-case character pair collates equally).

586 <small>Note: a future version of this standard may require these locales to define a collation sequence that  
587 has a total ordering of all characters (by changing "should" to "shall").</small>

588 <small>Note: users installing their own locales should ensure that they define a collation sequence with a  
589 total ordering of all characters unless an '@' modifier in the locale name (such as @icase) indicates that it  
590 has a special collation sequence.</small>

591 On Page: 149 Line: 4467 Section: 7.3.2.4 Collation Order  
592 (2013 edition Page: 150 Line: 4503)

593 Add a new paragraph and a small-font note:

594 Weights should be assigned such that the collation sequence has a total ordering of all characters unless an  
595 '@' modifier in the locale name indicates that it has a special collation sequence.

596 <small>Note: a future version of this standard may require a total ordering of all characters for  
597 implementation-provided locales that do not have an '@' modifier in the locale name. See [xref to  
598 7.3.2].</small>

599 On Page: 150 Line: 4481 Section: 7.3.2.4 Collation Order  
600 (2013 edition Page: 150 Line: 4517)

601 Change from:

602 Characters specified via an explicit or implicit UNDEFINED special symbol shall by default be assigned  
603 the same primary weight (that is, they belong to the same equivalence class).

604 to:

605 Characters specified via an explicit or implicit UNDEFINED special symbol shall by default be assigned  
606 the same primary weight (that is, they belong to the same equivalence class) if the collation order has more  
607 than one weight level. If the collation order has only one weight level, these characters should be assigned  
608 unique primary weights, equal to the relative order of their character in the character collation sequence,  
609 but may be assigned the same primary weight.

610 <small>Note: a future version of this standard may require these characters to be assigned unique primary  
611 weights if the collation order has only one weight level.</small>

612 On Page: 150 Line: 4503 Section: 7.3.2.4 Collation Order  
613 (2013 edition Page: 151 Line: 4539)

614 Delete the first entry in the example collation order:

615 UNDEFINED IGNORE; IGNORE

616 On Page: 150 Line: 4516 Section: 7.3.2.4 Collation Order  
617 (2013 edition Page: 151 Line: 4552)

618 Add the following as the last entry in the example collation order:

619 UNDEFINED IGNORE; ...

620 On Page: 151 Line: 4519 Section: 7.3.2.4 Collation Order  
 621 (2013 edition Page: 151 Line: 4555)

622 Delete item 1 in the numbered list:

623 The **UNDEFINED** means that all characters not specified in this definition (explicitly or via the ellipsis)  
 624 shall be ignored for collation purposes.

625 and renumber items 2-4 to be 1-3.

626 On Page: 151 Line: 4527 Section: 7.3.2.4 Collation Order  
 627 (2013 edition Page: 151 Line: 4563)

628 Add a new item 4 to the numbered list:

629 The **UNDEFINED** means that all characters not specified in this definition (explicitly or via the ellipsis)  
 630 shall be ignored when comparing primary weights, and have individual secondary weights based on their  
 631 ordinal encoded values.

632 *Rationale:* Austin Group Defect Report(s) applied: 938. See <http://austingroupbugs.net/view.php?id=938>.

633 **Change Number:** XBD/TC2/D4/0038 [663]

634 On Page: 151 Line: 4531 Section: 7.3.2.6 LC\_COLLATE Category in the POSIX Locale  
 635 (2013 edition Page: 151 Line: 4567)

636 Change from:

637 The collation sequence definition of the POSIX locale follows; the code listing depicts the *localedef* input.

638 to:

639 The minimum collation sequence definition of the POSIX locale follows; the code listing depicts the  
 640 *localedef* input. All characters not explicitly listed here shall be inserted in the character collation order  
 641 after the listed characters and shall be assigned unique primary weights. If the listed characters have ASCII  
 642 encoding, the other characters shall be in ascending order according to their coded character set values;  
 643 otherwise, the order of the other characters is unspecified. The collation sequence shall not include any  
 644 multi-character collating elements.

645 On Page: 151 Line: 4534 Section: 7.3.2.6 LC\_COLLATE Category in the POSIX Locale  
 646 (2013 edition Page: 151 Line: 4570)

647 Change from:

648 # This is the POSIX locale definition for the LC\_COLLATE category.  
 649 # The order is the same as in the ASCII codeset.

650 to:

651 # This is the minimum input for the POSIX locale definition for the  
 652 # LC\_COLLATE category. Characters in this list are in the same order  
 653 # as in the ASCII codeset.

654     *Rationale*: Austin Group Defect Report(s) applied: 663. See <http://austingroupbugs.net/view.php?id=663>.

655     **Change Number: XBD/TC2/D4/0039** [584]

656     On Page: 152 Line: 4582 Section: 7.3.2.6 LC\_COLLATE Category in the POSIX Locale

657     Change from:

658     <hyphen>

659     to:

660     <hyphen-minus>

661     *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

662     **Change Number: XBD/TC2/D4/0040** [912]

663     On Page: 163 Line: 5073 Section: 7.3.5.3 LC\_TIME Category in the POSIX Locale  
664     (2013 edition Page: 163 Line: 5109)

665     In the LC\_TIME category definition, change from:

666     <percent\_sign>

667     to:

668     <percent-sign>

669     *Rationale*: Austin Group Defect Report(s) applied: 912. See <http://austingroupbugs.net/view.php?id=912>.

670     **Change Number: XBD/TC2/D4/0041** [584]

671     On Page: 179 Line: 5728 Section: 8.3 Other Environment Variables

672     In the description of TZ change from:

673     <plus-sign> ('+') character, or the minus-sign

674     to:

675     <plus-sign> ('+') character, or the <hyphen-minus>

676     *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

677     **Change Number: XBD/TC2/D4/0042** [554]

678     On Page: 182 Line: 5842-5848 Section: 9.2 Regular Expression General Requirements

679     Change from:

680 In the regular expression processing described in POSIX.1-2008, the <newline> is regarded as an ordinary  
 681 character and both a <period> and a non-matching list can match one. The Shell and Utilities volume of  
 682 POSIX.1-2008 specifies within the individual descriptions of those standard utilities employing regular  
 683 expressions whether they permit matching of <newline> characters; if not stated otherwise, the use of  
 684 literal <newline> characters or any escape sequence equivalent produces undefined results.

685 to:

686 In the functions processing regular expressions described in the System Interfaces volume of POSIX.1-  
 687 2008, the <newline> is regarded as an ordinary character and both a <period> and a non-matching list can  
 688 match one. The Shell and Utilities volume of POSIX.1-2008 specifies within the individual descriptions of  
 689 those standard utilities employing regular expressions whether they permit matching of <newline>  
 690 characters; if not stated otherwise, the use of literal <newline> characters or any escape sequence  
 691 equivalent in either patterns or matched text produces undefined results .

692 **Change Number: XBD/TC2/D4/0043 [554]**

693 On Page: 183 Line: 5875 Section: 9.3.2 BRE Ordinary Characters

694 Change from:

695 The interpretation of an ordinary character preceded by a <backslash> ('\\') is undefined, except for:

696 to:

697 The interpretation of an ordinary character preceded by an unescaped <backslash> ('\\') is undefined, except  
 698 for:

699 On Page: 183 Line: 5887 Section: 9.3.3 BRE Special Characters

700 Change from:

701 that is not preceded by a <backslash>

702 to:

703 that is unescaped

704 On Page: 184 Line: 5894-5896 Section: 9.3.3 BRE Special Characters

705 Change from:

706 The <circumflex> shall be special when used as:

707 - An anchor (see Section XXX on page YYY)  
 708 - The first character of a bracket expression (see Section AAA on page BBB)

709 to:

710 The <circumflex> shall be special when used as an anchor (see Section XXX, on page YYY). The  
 711 <circumflex> shall signify a non matching list expression when it occurs first in a list, immediately  
 712 following a <left-square-bracket> (see Section AAA on page BBB).

713 *Rationale*: Austin Group Defect Report(s) applied: 554. See <http://austingroupbugs.net/view.php?id=554>.

714 **Change Number: XBD/TC2/D4/0044** [938]

715 On Page: 184 Line: 5902 Section: 9.3.5 RE Bracket Expression  
716 (2013 edition Page: 184 Line: 5944)

717 Change from:

718 A bracket expression (an expression enclosed in square brackets, "[ ]") is an RE that shall match a single  
719 collating element contained in the non-empty set of collating elements represented by the bracket  
720 expression.

721 to:

722 A bracket expression (an expression enclosed in square brackets, "[ ]") is an RE that shall match a specific  
723 set of single characters, and may match a specific set of multi-character collating elements, based on the  
724 non-empty set of list expressions contained in the bracket expression.

725 On Page: 184 Line: 5907 Section: 9.3.5 RE Bracket Expression  
726 (2013 edition Page: 184 Line: 5949)

727 In list item 1, change from:

728 It consists of one or more expressions: collating elements, collating symbols, ...

729 to:

730 It consists of one or more expressions: ordinary characters, collating elements, collating symbols, ...

731 On Page: 184 Line: 5921 Section: 9.3.5 RE Bracket Expression  
732 (2013 edition Page: 184 Line: 5963)

733 In list item 2, change from:

734 A matching list expression specifies a list that shall match any single-character collating element in any of  
735 the expressions represented in the list. The first character in the list shall not be the <circumflex>; for  
736 example, "[abc]" is an RE that matches any of the characters 'a', 'b', or 'c'.

737 to:

738 A matching list expression specifies a list that shall match any single character that is matched by one of  
739 the expressions represented in the list. The first character in the list can not be the <circumflex>. An  
740 ordinary character in the list should only match that character, but may match any single character that  
741 collates equally with that character; for example, "[abc]" is an RE that should only match one of the  
742 characters 'a', 'b', or 'c'.

743 <small>Note: a future version of this standard may require that an ordinary character in the list only  
744 matches that character.</small>

745 *Rationale*: Austin Group Defect Report(s) applied: 938. See <http://austingroupbugs.net/view.php?id=938>.

746 **Change Number: XBD/TC2/D4/0045** [872]

747 On Page: 184 Line: 5926 Section: 9.3.5 RE Bracket Expression  
 748 (2013 edition Page: 184 Line: 5968)

749 In list item 3, change from:

750 A non-matching list expression begins with a <circumflex> ('^'), and specifies a list that shall match any  
 751 single-character collating element except for the expressions represented in the list after the leading  
 752 <circumflex>.

753 to:

754 A non-matching list expression begins with a <circumflex> ('^'), and the matching behavior shall be the  
 755 logical inverse of the corresponding matching list expression (the same bracket expression but without the  
 756 leading <circumflex>).

757 *Rationale:* Austin Group Defect Report(s) applied: 872. See <http://austingroupbugs.net/view.php?id=872>.

758 **Change Number: XBD/TC2/D4/0046** [938]

759 On Page: 184 Line: 5928 Section: 9.3.5 RE Bracket Expression  
 760 (2013 edition Page: 184 Line: 5970)

761 In list item 3, change from:

762 For example, "[^abc]" is an RE that matches any character except the characters 'a', 'b', or 'c'.

763 to:

764 For example, if the RE "[abc]" only matches 'a', 'b', or 'c', then "[^abc]" is an RE that matches any character  
 765 except 'a', 'b', or 'c'.

766 On Page: 185 Line: 5953 Section: 9.3.5 RE Bracket Expression  
 767 (2013 edition Page: 185 Line: 5995)

768 In list item 6, change from:

769 The set of single-character collating elements whose characters belong to the character class

770 to:

771 The set of single characters that belong to the character class

772 *Rationale:* Austin Group Defect Report(s) applied: 938. See <http://austingroupbugs.net/view.php?id=938>.

773 **Change Number: XBD/TC2/D4/0047** [584]

774 On Page: 185 Line: 5972 Section: 9.3.5 RE Bracket Expression

775 Change from:

776 <hyphen>

777 to:

778 <hyphen-minus>

779 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

780 **Change Number: XBD/TC2/D4/0048** [584]

781 On Page: 186 Line: 5982,5989,5992 Section: 9.3.5 RE Bracket Expression

782 Change from:

783 <hyphen>

784 to:

785 <hyphen-minus>

786 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

787 **Change Number: XBD/TC2/D4/0049** [595]

788 On Page: 186 Line: 6016 Section: 9.3.6 BREs Matching Multiple Characters

789 Change rule 3 from:

790 For example, the expression " $^{\wedge}(.*)\backslash1\$$ " matches lines consisting of two adjacent appearances of the same  
791 string,

792 to:

793 For example, the expression " $^{\wedge}(.*)\backslash1\$$ " matches strings consisting of two adjacent appearances of the  
794 same substring,

795 On Page: 187 Line: 6053 Section: 9.3.8 BRE Expression Anchoring

796 Change from:

797 A BRE can be limited to matching strings that begin or end a line; this is called "anchoring".

798 to:

799 A BRE can be limited to matching expressions that begin or end a string; this is called "anchoring".

800 *Rationale:* Austin Group Defect Report(s) applied: 595. See <http://austingroupbugs.net/view.php?id=595>.

801

802 **Change Number: XBD/TC2/D4/0050** [554]

803 On Page: 188 Line: 6082-6083 Section: 9.4.2 ERE Ordinary Characters

804 Change from:

805 The interpretation of an ordinary character preceded by a <backslash> ('\\') is undefined.

806 to:

807 The interpretation of an ordinary character preceded by an unescaped <backslash> ('\\') is undefined, except  
 808 in the context of a bracket expression (see XREF to ERE bracket expression).

809 On Page: 188 Line: 6092 Section: 9.4.3 ERE Special Characters

810 Add the following text after line 6092:

811 A <left-square-bracket> that is unescaped and is not part of a bracket expression also produces undefined  
 812 results.

813 On Page: 188 Line: 6098-6099 Section: 9.4.3 ERE Special Characters

814 Change from:

815 If these characters appear first in an ERE, or immediately following a <vertical-line>, <circumflex>, or  
 816 <left-parenthesis>

817 to:

818 If these characters appear first in an ERE, or immediately following an unescaped <vertical-line>,  
 819 <circumflex>, <dollar-sign>, or <left-parenthesis>

820 On Page: 189 Line: 6106-6109 Section: 9.4.3 ERE Special Characters

821 Change from:

822 The <circumflex> shall be special when used as:  
 823 - An anchor (see Section XXX on page YYY)  
 824 - The first character of a bracket expression (see Section AAA on page BBB)

825 to:

826 The <circumflex> shall be special when used as an anchor (see Section XXX, on page YYY). The  
 827 <circumflex> shall signify a non matching list expression when it occurs first in a list, immediately  
 828 following a <left-square-bracket> (see Section AAA on page BBB).

829 *Rationale:* Austin Group Defect Report(s) applied: 554. See <http://austingroupbugs.net/view.php?id=554>.

830

831 **Change Number: XBD/TC2/D4/0051** [595]

832 On Page: 190 Line: 6175 Section: 9.4.9 ERE Expression Anchoring

833 Change from:

834 An ERE can be limited to matching strings that begin or end a line; this is called "anchoring".

835 to:

836 An ERE can be limited to matching expressions that begin or end a string; this is called "anchoring".

837 *Rationale:* Austin Group Defect Report(s) applied: 595. See <http://austingroupbugs.net/view.php?id=595>.

838 **Change Number: XBD/TC2/D4/0052** [554]

839 On Page: 192 Line: 6233 Section: 9.5.1 BRE/ERE Grammar Lexical Conventions

840 For token SPEC\_CHAR delete the phrase:

841 or when first in a bracket expression

842 On Page: 195 Line: 6379-6380 Section: 9.5.3 ERE Grammar

843 Change from:

844 The ERE grammar does not permit several constructs that previous sections specify as having undefined results:

846 to:

847 The ERE grammar does not permit several constructs that previous sections specify as having undefined results. Additionally, there are some constructs which the grammar permits but which still give undefined results:

850 On Page: 195 Line: 6381 Section: 9.5.3 ERE Grammar

851 Change from:

852 ORD\_CHAR preceded by a <backslash> character

853 to:

854 ORD\_CHAR preceded by an unescaped <backslash> character

855 On Page: 195 Line: 6383 Section: 9.5.3 ERE Grammar

856 Add '\$' to the list of characters.

857 *Rationale:* Austin Group Defect Report(s) applied: 554. See <http://austingroupbugs.net/view.php?id=554>.

858 **Change Number: XBD/TC2/D4/0053** [916]

859 On Page: 195 Line: 6387 Section: 9.5.3 ERE Grammar  
860 (2013 edition Page: 195 Line: 6433)

861 Change from:

862 Conforming applications cannot use such constructs.

863 to:

864 Strictly conforming applications cannot use such constructs.

865 *Rationale:* Austin Group Defect Report(s) applied: 916. See <http://austingroupbugs.net/view.php?id=916>.

866 **Change Number: XBD/TC2/D4/0054** [967]

867 On Page: 198 Line: 6438-6454 Section: 10.2 Output Devices and Terminal Types  
868 (2013 edition Page: 198 Line: 6484-6500)

869 Delete the two **Symbolic Name** columns from Table 10-1.

870 *Rationale:* Austin Group Defect Report(s) applied: 967. See <http://austingroupbugs.net/view.php?id=967>.

871 **Change Number: XBD/TC2/D4/0055** [745]

872 On Page: 204 Line: 6682,6684 Section: 11.1.9 Special Characters

873 On line 6682, for CR change from:

874 carriage-return

875 to:

876 <carriage-return>

877 On line 6684, append a sentence to the description of CR:

878 It cannot be changed.

879 *Rationale:* Austin Group Defect Report(s) applied: 745. See <http://austingroupbugs.net/view.php?id=745>.

880 **Change Number: XBD/TC2/D4/0056** [584]

881 On Page: 213 Line: 7006 Section: 12.1 Utility Argument Syntax

882 Change from:

883 <hyphen>

884 to:

885 <hyphen-minus>

886 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

887 **Change Number: XBD/TC2/D4/0057** [813]

888 On Page: 214 Line: 7056 Section: 12.1 Utility Argument Syntax

889 After the third bullet item, add a new bullet item:

890     • When the utility description states that the number is a file size-related value (such as a file size or  
891        offset, line number, or block count), numerals in the range 0 to the maximum file size supported  
892        by the implementation are syntactically recognized as numeric values (see XCU 1.5  
893        Considerations for Utilities in Support of Files of Arbitrary Size). Where negative values are  
894        permitted, any value in the range -(maximum file size) to the maximum file size is accepted.

895 *Rationale*: Austin Group Defect Report(s) applied: 813. See <http://austingroupbugs.net/view.php?id=813>.  
896 The normative requirement in XCU 1.5 does not match the syntax utility guidelines which suggest very  
897 large values may be supported but are not required to be.

898 **Change Number: XBD/TC2/D4/0058** [579]

899 On Page: 220 Line: 7211 Section: <aio.h>

900 In the DESCRIPTION section, change from:

901 The tag **sigevent** shall be declared as naming an incomplete structure type, the contents of which are  
902 described in the <**signal.h**> header.

903 to:

904 The <**aio.h**> header shall define the **sigevent** structure and **sigval** union as described in <**signal.h**>.

905 *Rationale*: Austin Group Defect Report(s) applied: 579. See <http://austingroupbugs.net/view.php?id=579>.

906 **Change Number: XBD/TC2/D4/0059** [496]

907 On Page: 236 Line: 7737 Section: <errno.h>

908 In the DESCRIPTION section, change from:

909 [ENOSYS] Function not supported.

910 to:

911 [ENOSYS] Functionality not supported.

912 *Rationale*: Austin Group Defect Report(s) applied: 496. See <http://austingroupbugs.net/view.php?id=496>.

913 **Change Number: XBD/TC2/D4/0060** [847]

914 On Page: 240 Line: 7878 Section: <fcntl.h>  
 915 (2013 edition Page: 238 Line: 7885)

916 In the DESCRIPTION section, change from:

917 O\_DIRECTORY Fail if not a directory.

918 to:

919 O\_DIRECTORY Fail if file is a non-directory file.

920 *Rationale:* Austin Group Defect Report(s) applied: 847. See <http://austingroupbugs.net/view.php?id=847>.

921 **Change Number: XBD/TC2/D4/0061** [666]

922 On Page: 270 Line: 8853-8856 Section: <limits.h>

923 In the DESCRIPTION section, delete:

924 {RE\_DUP\_MAX}

925 Maximum number of repeated occurrences of a BRE or ERE interval expression; see Section 9.3.6  
 926 (on page 186) and Section 9.4.6 (on page 189).

927 Minimum Acceptable Value: {\_POSIX2\_RE\_DUP\_MAX}

928 On Page: 273 Line: 9001-9003 Section: <limits.h>

929 Change from:

930 {RE\_DUP\_MAX}

931 Maximum number of repeated occurrences of a regular expression permitted when using the interval  
 932 notation  $\{\{m,n\}\}$ ; see Chapter 9 (on page 181).

933 to (copied from line 8854):

934 {RE\_DUP\_MAX}

935 Maximum number of repeated occurrences of a BRE or ERE interval expression; see Section 9.3.6  
 936 (on page 186) and Section 9.4.6 (on page 189).

937 On Page: 273 Line: 9004 Section: <limits.h>

938 Change from:

939 {\_POSIX2\_RE\_DUP\_MAX}

940 to:

941 {\_POSIX\_RE\_DUP\_MAX}

942 On Page: 275 Line: 9076-9077 Section: <limits.h>

943 Change from:

944 The number of repeated occurrences of a BRE permitted by the *regexec()* and *regcomp()* functions when  
945 using the interval notation  $\{(m,n)\}$ ; see Section 9.3.6 (on page 186).

946 to (copied from line 8854):

947 Maximum number of repeated occurrences of a BRE or ERE interval expression; see Section 9.3.6 (on  
948 page 186) and Section 9.4.6 (on page 189).

949 On Page: 277 Line: 9174-9175 Section: <limits.h>

950 Change from:

951 Maximum number of repeated occurrences of a regular expression permitted when using the interval  
952 notation  $\{m,n\}$ ; see Chapter 9 (on page 181).

953 to (copied from line 8854):

954 Maximum number of repeated occurrences of a BRE or ERE interval expression; see Section 9.3.6 (on  
955 page 186) and Section 9.4.6 (on page 189).

956 *Rationale*: Austin Group Defect Report(s) applied: 666. See <http://austingroupbugs.net/view.php?id=666>.

957 **Change Number: XBD/TC2/D4/0062** [781]

958 On Page: 283 Line: 9438 Section: <locale.h>  
959 (2013 edition Page: 286 Line: 9521)

960 In the DESCRIPTION section, change from:

961 Implementations may add additional masks using the form *LC\_\** and an uppercase letter.

962 to:

963 Additional macro definitions, beginning with the characters *LC\_* and an uppercase letter, may also be  
964 specified by the implementation.

965 On Page: 284 Line: 9447 Section: <locale.h>  
966 (2013 edition Page: 287 Line: 9530)

967 Delete:

968 [CX]Implementations may add additional masks using the form *LC\_\***MASK*.[/CX]

969 On Page: 284 Line: 9464 Section: <locale.h>  
970 (2013 edition Page: 287 Line: 9547)

971 In the RATIONALE section, change from:

972 None.

973 to:

974 It is suggested that each category macro name for use in *setlocale()* have a corresponding macro name  
 975 ending in *\_MASK* for use in *newlocale()*.

976 *Rationale*: Austin Group Defect Report(s) applied: 781. See <http://austingroupbugs.net/view.php?id=781>.

977 **Change Number: XBD/TC2/D4/0063** [801]

978 On Page: 286 Line: 9512 Section: <math.h>  
 979 (2013 edition Page: 289 Line: 9596)

980 In the DESCRIPTION section, change from:

981 shall be accurate within the precision of the **double** type

982 to:

983 shall be accurate to at least the precision of the **double** type

984 *Rationale*: Austin Group Defect Report(s) applied: 801. See <http://austingroupbugs.net/view.php?id=801>.

985 **Change Number: XBD/TC2/D4/0064** [801]

986 On Page: 292 Line: 9776 Section: <math.h>  
 987 (2013 edition Page: 295 Line: 9860)

988 Add a note to the APPLICATION USAGE section:

989 Note that if **FLT\_EVAL\_METHOD** is neither 0 nor 1, then some constants might  
 990 not compare equal as expected, for example `(double)M_PI == M_PI` can fail.

991 *Rationale*: Austin Group Defect Report(s) applied: 801. See <http://austingroupbugs.net/view.php?id=801>.

992 **Change Number: XBD/TC2/D4/0065** [934]

993 On Page: 303 Line: 10154 Section: <netinet/in.h>  
 994 (2013 edition Page: 306 Line: 10238)

995 In the DESCRIPTION section, change from:

996 The **sockaddr\_in6** structure shall be set to zero by an application prior to using it, since implementations  
 997 are free to have additional, implementation-defined fields in **sockaddr\_in6**.

998 to:

999 Prior to calling a function in this standard which reads values from a **sockaddr\_in6** structure (for example,  
 1000 *bind()* or *connect()*), the application shall ensure that all members of the structure, including any additional  
 1001 non-standard members, if any, are initialized. If the **sockaddr\_in6** structure has a non-standard member,  
 1002 and that member has a value other than the value that would result from default initialization, the behavior  
 1003 of any function in this standard that reads values from the **sockaddr\_in6** structure is implementation-

1004 defined. All functions in this standard that return data in a **sockaddr\_in6** structure (for example,  
1005 *getaddrinfo()* or *accept()*) shall initialize the structure in a way that meets the above requirements, and shall  
1006 ensure that each non-standard member, if any, has a value that produces the same behavior as default  
1007 initialization would in all functions in this standard which read values from a **sockaddr\_in6** structure.

1008 *Rationale*: Austin Group Defect Report(s) applied: 934. See <http://austingroupbugs.net/view.php?id=934>.

1009 **Change Number: XBD/TC2/D4/0066** [952]

1010 On Page: 304 Line: 10185 Section: <netinet/in.h>  
1011 (2013 edition Page: 307 Line: 10269)

1012 In the DESCRIPTION section, change from:

1013 The <netinet/in.h> header shall define the following symbolic constants for use as destination addresses  
1014 for *connect()*, *sendmsg()*, and *sendto()*:

1015 INADDR\_ANY IPv4 local host address.

1016 INADDR\_BROADCAST IPv4 broadcast address.

1017 to:

1018 The <netinet/in.h> header shall define the following symbolic constant for use as a local address in the  
1019 structure passed to *bind()*:

1020 INADDR\_ANY IPv4 wildcard address.

1021 The <netinet/in.h> header shall define the following symbolic constant for use as a destination address in  
1022 the structures passed to *connect()*, *sendmsg()*, and *sendto()*:

1023 INADDR\_BROADCAST IPv4 broadcast address.

1024 *Rationale*: Austin Group Defect Report(s) applied: 952. See <http://austingroupbugs.net/view.php?id=952>.

1025 **Change Number: XBD/TC2/D4/0067** [934]

1026 On Page: 306 Line: 10239 Section: <netinet/in.h>  
1027 (2013 edition Page: 309 Line: 10323)

1028 In the APPLICATION USAGE section, change from:

1029 None.

1030 to:

1031 Although applications are required to initialize all members (including any non-standard ones) of a  
1032 **sockaddr\_in6** structure, the same is not required for the **sockaddr\_in** structure, since historically many  
1033 applications only initialized the standard members. Despite this, applications are encouraged to initialize  
1034 **sockaddr\_in** structures in a manner similar to the required initialization of **sockaddr\_in6** structures.

1035 Although it is common practice to initialize a **sockaddr\_in6** structure using:

1036 `struct sockaddr_in6 sa;`  
 1037 `memset(&sa, 0, sizeof sa);`

1038 this method is not portable according to this standard, because the structure can contain pointer or floating  
 1039 point members that are not required to have an all-bits-zero representation after default initialization.  
 1040 Portable methods make use of default initialization, for example:

1041 `struct sockaddr_in6 sa = { 0 };`

1042 or

1043 `static struct sockaddr_in6 sa_init;`  
 1044 `struct sockaddr_in6 sa = sa_init;`

1045 A future version of this standard may require that a pointer object with an all-bits-zero representation is a  
 1046 null pointer, and that **sockaddr\_in6** does not have any floating-point members if a floating-point object  
 1047 with an all-bits-zero representation does not have the value 0.0.

1048 *Rationale:* Austin Group Defect Report(s) applied: 934. See <http://austingroupbugs.net/view.php?id=934>.

1049 **Change Number: XBD/TC2/D4/0068** [952]

1050 On Page: 306 Line: 10241 Section: <netinet/in.h>  
 1051 (2013 edition Page: 309 Line: 10325)

1052 In the RATIONALE section, change from:

1053 None.

1054 to:

1055 The INADDR\_ANY and INADDR\_BROADCAST values are byte-order-neutral and thus their byte order  
 1056 is not specified. Many implementations have additional constants as extensions, such as  
 1057 INADDR\_LOOPBACK, that are not byte-order-neutral. Traditionally, these constants are in host byte  
 1058 order, requiring the use of `htonl()` when using them in a **sockaddr\_in** structure.

1059 *Rationale:* Austin Group Defect Report(s) applied: 952. See <http://austingroupbugs.net/view.php?id=952>.

1060 **Change Number: XBD/TC2/D4/0069** [624]

1061 On Page: 312 Line: 10448-10449 Section: <pthread.h>

1062 In the DESCRIPTION section, delete `pthread_cleanup_push` and `pthread_cleanup_pop` from the list of  
 1063 functions.

1064 Add a new paragraph after line 10542:

1065

1066 The following may be declared as functions, or defined as macros, or both.  
 1067 If functions are declared, function prototypes shall be provided.

1068

1069 *pthread\_cleanup\_pop*  
1070 *pthread\_cleanup\_push*

1071 *Rationale*: Austin Group Defect Report(s) applied: 624. See <http://austingroupbugs.net/view.php?id=624>.

1072 **Change Number: XBD/TC2/D4/0070** [536]

1073 On Page: 329 Line: 11005-11006 Section: <signal.h>

1074 In the DESCRIPTION section, change from:

1075 [CX]The ISO C standard only requires the signal names SIGABRT, SIGFPE, SIGILL, SIGINT,  
1076 SIGSEGV, and SIGTERM to be defined.[/CX]

1077 to:

1078 The ISO C standard only requires the signal names SIGABRT, SIGFPE, SIGILL, SIGINT, SIGSEGV, and  
1079 SIGTERM to be defined. An implementation need not generate any of these six signals, except as a result  
1080 of explicit use of interfaces that generate signals, such as *raise()*, [CX]*kill()*, the General Terminal Interface  
1081 (see section 11.1.9), and the *kill* utility, unless otherwise stated (see for example XSH 2.8.3.3)[/CX].

1082 *Rationale*: Austin Group Defect Report(s) applied: 536. See <http://austingroupbugs.net/view.php?id=536>.

1083 **Change Number: XBD/TC2/D4/0071** [690]

1084 On Page: 331 Line: 11073 Section: <signal.h>  
1085 (2013 edition Page: 334 Line: 11158)

1086 In the DESCRIPTION section change from:

1087 [CX]SA\_NOCLDWAIT Causes implementations not to create zombie processes on child death.[/CX]

1088 to:

1089 [XSI]SA\_NOCLDWAIT Causes implementations not to create zombie processes or status information on  
1090 child termination. See *sigaction()*, page 1932.[/XSI]

1091 (Note the change from CX shading to XSI shading.)

1092 *Rationale*: Austin Group Defect Report(s) applied: 690. See <http://austingroupbugs.net/view.php?id=690>.

1093 **Change Number: XBD/TC2/D4/0072** [594]

1094 On Page: 334 Line: 11164 Section: <signal.h>

1095 In the DESCRIPTION section, in the "Value" column for "**int si\_status**" change from:

1096 Exit value or signal.

1097 to:

1098 If *si\_code* is equal to CLD\_EXITED, then *si\_status* holds the exit value of the process; otherwise, it is  
 1099 equal to the signal that caused the process to change state. The exit value in *si\_status* shall be equal to the  
 1100 full exit value (that is, the value passed to *\_exit()*, *\_Exit()* or *exit()*, or returned from *main()*); it shall not be  
 1101 limited to the least significant eight bits of the value.

1102 *Rationale*: Austin Group Defect Report(s) applied: 594. See <http://austingroupbugs.net/view.php?id=594>.  
 1103 The standard is unclear about when *si\_status* contains an exit status and when it contains a signal.

1104 **Change Number: XBD/TC2/D4/0073** [844]

1105 On Page: 334 Line: 11193 Section: <signal.h>  
 1106 (2013 edition Page: 337 Line: 11278)

1107 In the DESCRIPTION section, change from:

1108 `int sigqueue(pid_t, int, const union sigval);`

1109 to:

1110 `int sigqueue(pid_t, int, union sigval);`

1111 *Rationale*: Austin Group Defect Report(s) applied: 844. See <http://austingroupbugs.net/view.php?id=844>.

1112 **Change Number: XBD/TC2/D4/0074** [536]

1113 On Page: 335 Line: 11213 Section: <signal.h>

1114 In the SEE ALSO section, add:

1115 XCU *kill*

1116 *Rationale*: Austin Group Defect Report(s) applied: 536. See <http://austingroupbugs.net/view.php?id=536>.

1117 **Change Number: XBD/TC2/D4/0075** [663]

1118 On Page: 355 Line: 11953 Section: <stdlib.h>  
 1119 (2013 edition Page: 358 Line: 12042)

1120 After:

1121 {MB\_CUR\_MAX} Maximum number of bytes in a character specified by the current locale (category  
 1122 LC\_CTYPE).

1123 add a new sentence:

1124 [CX]In the POSIX locale the value of {MB\_CUR\_MAX} shall be 1.[/CX]

1125 *Rationale*: Austin Group Defect Report(s) applied: 663. See <http://austingroupbugs.net/view.php?id=663>.  
 1126 The intention was always that the POSIX locale should have an 8-bit-clean single-byte encoding. The  
 1127 omission of an explicit statement to that effect was an oversight.

1128 **Change Number: XBD/TC2/D4/0076** [801]

1129 On Page: 365 Line: 12336 Section: <stropts.h>  
1130 (2013 edition Page: 368 Line: 12425)

1131 In the DESCRIPTION section, add a new paragraph before the text "The following shall be declared...":

1132 The <stropts.h> header may also define macros for message types using names that start with M\_.

1133 *Rationale:* Austin Group Defect Report(s) applied: 801. See <http://austingroupbugs.net/view.php?id=801>.

1134 **Change Number: XBD/TC2/D4/0077** [934]

1135 On Page: 385 Line: 12911 Section: <sys/socket.h>  
1136 (2013 edition Page: 388 Line: 13000)

1137 In the DESCRIPTION section, add a new paragraph:

1138 The value of AF\_UNSPEC shall be 0.

1139 *Rationale:* Austin Group Defect Report(s) applied: 934. See <http://austingroupbugs.net/view.php?id=934>.  
1140 Historically some applications initialize just the standard members of sockaddr\_in, and some initialize the  
1141 whole structure (using default initialization or memset()). There may be applications which do the latter  
1142 and then rely on the zero value of the sin\_family member being set to AF\_UNSPEC. Therefore the  
1143 standard should require that AF\_UNSPEC has the value 0.

1144 **Change Number: XBD/TC2/D4/0078** [531]

1145 On Page: 390 Line: 13129 Section: <sys/stat.h>

1146 At the end of the DESCRIPTION section, add:

1147 Inclusion of the <sys/stat.h> header may make visible all symbols from the <time.h> header.

1148 *Rationale:* Austin Group Defect Report(s) applied: 531. See <http://austingroupbugs.net/view.php?id=531>.

1149 **Change Number: XBD/TC2/D4/0079** [856]

1150 On Page: 399 Line: 13386 Section: <sys/types.h>  
1151 (2013 edition Page: 402 Line: 13477)

1152 In the DESCRIPTION section, remove the XSI shading from the line:

1153 **suseconds\_t** Used for time in microseconds.

1154 *Rationale:* Austin Group Defect Report(s) applied: 856. See <http://austingroupbugs.net/view.php?id=856>.

1155

1156 **Change Number: XBD/TC2/D4/0080** [659]

1157 On Page: 399 Line: 13408 Section: <sys/types.h>

1158 In the DESCRIPTION section:

1159 Add **timer\_t** to the list of exceptions to the arithmetic types requirement.

1160 On Page: 400 Line: 13438 Section: <sys/types.h>

1161 In the DESCRIPTION section:

1162 Add **timer\_t** to the list of types that have no comparison or assignment operators.

1163 *Rationale:* Austin Group Defect Report(s) applied: 659. See <http://austingroupbugs.net/view.php?id=659>.

1164 **Change Number: XBD/TC2/D4/0081** [934]

1165 On Page: 403 Line: 13525 Section: <sys/un.h>

1166 (2013 edition Page: 406 Line: 13620)

1167 In the APPLICATION USAGE section, add a new paragraph:

1168 Although applications are required to initialize all members (including any non-standard ones) of a **sockaddr\_in6** structure (see <netinet/in.h>), the same is not required for the **sockaddr\_un** structure, since historically many applications only initialized the standard members. Despite this, applications are encouraged to initialize **sockaddr\_un** structures in a manner similar to the required initialization of **sockaddr\_in6** structures.

1173 *Rationale:* Austin Group Defect Report(s) applied: 934. See <http://austingroupbugs.net/view.php?id=934>.

1174 **Change Number: XBD/TC2/D4/0082** [579]

1175 On Page: 405 Line: 13598 Section: <sys/wait.h>

1176 In the DESCRIPTION section, change from:

1177 The <sys/wait.h> header shall define the **siginfo\_t** type as described in <signal.h>.

1178 to:

1179 The <sys/wait.h> header shall define the **siginfo\_t** type and the **sigval** union as described in <signal.h>.

1180 *Rationale:* Austin Group Defect Report(s) applied: 579. See <http://austingroupbugs.net/view.php?id=579>.

1181 **Change Number: XBD/TC2/D4/0083** [564]

1182 On Page: 406 Line: 13621 Section: <sys/wait.h>

1183 In the CHANGE HISTORY section, add a new paragraph:

1184 The requirement for `<sys/wait.h>` to define the **rusage** structure as described in `<sys/resource.h>` is  
 1185 removed, and `<sys/wait.h>` is no longer allowed to make visible all symbols from `<sys/resource.h>`.

1186 *Rationale:* Austin Group Defect Report(s) applied: 564. See <http://austingroupbugs.net/view.php?id=564>.

1187 **Change Number: XBD/TC2/D4/0084** [707]

1188 On Page: 409 Line: 13695-13698 Section: `<tar.h>`  
 1189 (2013 edition Page: 412 Line: 13791-13794)

1190 In the DESCRIPTION section, replace the entries in the "General Definitions" table with:

TMAGIC	"ustar"	Used in the <i>magic</i> field in the <b>ustar</b> header block, including the trailing null byte
TMAGLEN	6	Length in octets of the <i>magic</i> field
TVERSION	"00"	Used in the <i>version</i> field in the <b>ustar</b> header block, excluding the trailing null byte
TVERSLEN	2	Length in octets of the <i>version</i> field

1191

1192 *Rationale:* Austin Group Defect Report(s) applied: 707. See <http://austingroupbugs.net/view.php?id=707>.

1193 **Change Number: XBD/TC2/D4/0085** [783]

1194 On Page: 431 Line: 14489 Section: `<unistd.h>`

1195 In the DESCRIPTION section:

1196 Add IP6 margin marker and shading to `_POSIX_IPV6` (lines 14489-14492)

1197 *Rationale:* Austin Group Defect Report(s) applied: 783. See <http://austingroupbugs.net/view.php?id=783>.

1198 **Change Number: XBD/TC2/D4/0086** [911]

1199 On Page: 437 Line: 14773 Section: `<unistd.h>`  
 1200 (2013 edition Page: 440 Line: 14878)

1201 In the DESCRIPTION section, change from:

1202 This is the value for the *PATH* environment variable that finds all standard utilities.

1203 to:

1204 This is the value for the *PATH* environment variable that finds all of the standard utilities that are provided  
 1205 in a manner accessible via the *exec* family of functions.

1206 *Rationale:* Austin Group Defect Report(s) applied: 911. See <http://austingroupbugs.net/view.php?id=911>.

1207 **Change Number: XBD/TC2/D4/0087** [566]

1208 On Page: 443 Line: 15037 Section: <unistd.h>

1209 In the DESCRIPTION section, change from:

1210 **intptr\_t** type as described in <inttypes.h>.

1211 to:

1212 **intptr\_t** type as described in <stdint.h>.

1213 On Page: 445 Line: 15127 Section: <unistd.h>

1214 In the DESCRIPTION section, add a new paragraph:

1215 Inclusion of the <unistd.h> header may make visible all symbols from the headers <stddef.h>, <stdint.h>,  
1216 and <stdio.h>.

1217 *Rationale:* Austin Group Defect Report(s) applied: 566. See <http://austingroupbugs.net/view.php?id=566>.

1218 **Change Number: XBD/TC2/D4/0088** [73]

1219 On Page: 454 Line: 15481-15482 Section: <wchar.h>

1220 In the DESCRIPTION section, change from:

1221 The following shall be declared as functions and may also be defined as macros. Function prototypes shall  
1222 be provided for use with ISO C standard compilers.

1223 to:

1224 The following shall be declared as functions and may also be defined as macros. Function prototypes shall  
1225 be provided for use with ISO C standard compilers. Arguments to functions in this list can point to arrays  
1226 containing **wchar\_t** values that do not correspond to members of the character set of the current locale.  
1227 Such values shall be processed according to the specified semantics, unless otherwise stated.

1228 *Rationale:* Austin Group Defect Report(s) applied: 73. See <http://austingroupbugs.net/view.php?id=73>.

1229

1230 A clarification has been made in the C11 standard.

1231

1232 **3. Changes to System Interfaces**

1233 This section contains the set of changes to the text of the System Interfaces.  
1234 [Note to reviewers: References to defect reports are provided to aid reviewers.]

1235 **Change Number: XSH/TC2/D4/0001** [801]

1236 On Page: 471 Line: 15977 Section: 2.2.2 The Name Space  
1237 (2013 edition Page: 475 Line: 16085)

1238 Add an M\_ prefix reservation for <math.h> in the table, shaded XSI.

1239 *Rationale:* Austin Group Defect Report(s) applied: 801. See <http://austingroupbugs.net/view.php?id=801>.

1240 **Change Number: XSH/TC2/D4/0002** [780]

1241 On Page: 471 Line: 15991-15992 Section: 2.2.2 The Name Space

1242 Add CX shading to the two unshaded lines in the <signal.h> table row.

1243 *Rationale:* Austin Group Defect Report(s) applied: 780. See <http://austingroupbugs.net/view.php?id=780>.

1244 **Change Number: XSH/TC2/D4/0003** [790]

1245 On Page: 471 Line: 15997 Section: 2.2.2 The Name Space

1246 Remove the <stdint.h> row from the table and append the following sentence to the small-font note after  
1247 the table:

1248 The C standard reserves int[0-9a-z]\*\_t and uint[0-9a-z]\*\_t in <stdint.h>; this is not included in the table  
1249 above because it is covered by the reserved \_t suffix for any header.

1250 *Rationale:* Austin Group Defect Report(s) applied: 790. See <http://austingroupbugs.net/view.php?id=790>.  
1251 These patterns are redundant as \_t is a reserved suffix.

1252 **Change Number: XSH/TC2/D4/0004** [780]

1253 On Page: 472 Line: 16025-16035 Section: 2.2.2 The Name Space

1254 On lines 16025-16027 add CX shading to the <time.h> table row.

1255 On line 16029 add OB shading to the <utime.h> table row.

1256 On line 16035 add CX shading to the **ANY header** table row.

1257 *Rationale:* Austin Group Defect Report(s) applied: 780. See <http://austingroupbugs.net/view.php?id=780>.

1258

1259 **Change Number: XSH/TC2/D4/0005** [790]

1260 On Page: 472 Line: 16036 Section: 2.2.2 The Name Space

1261 Change from:

1262 The notation [A-Z] indicates any uppercase letter in the portable character set.

1263 to:

1264 The notation [0-9] indicates any digit. The notation [A-Z] indicates any uppercase letter in the portable  
 1265 character set.

1266 *Rationale:* Austin Group Defect Report(s) applied: 790. See <http://austingroupbugs.net/view.php?id=790>.

1267 **Change Number: XSH/TC2/D4/0006** [782]

1268 On Page: 472 Line: 16038 Section: 2.2.2 The Name Space  
 1269 (2013 edition Page: 476 Line: 16146)

1270 Add a new paragraph (not as part of the preceding Note):

1271 Implementations may also add symbols to the `<complex.h>` header with the following complete names or  
 1272 the same names suffixed with f or l:

1273 cerf cexpml clog2

cerfc clog10 clgamma

cexp2 clog1p ctgamma

1274 *Rationale:* Austin Group Defect Report(s) applied: 782. See <http://austingroupbugs.net/view.php?id=782>.

1275 **Change Number: XSH/TC2/D4/0007** [790]

1276 On Page: 473 Line: 16069 Section: 2.2.2 The Name Space

1277 Change from:

1278 The notation [0-9] indicates any digit. The notation [A-Z] indicates any uppercase letter in the portable  
 1279 character set. The notation [0-9a-z\_] indicates any digit, any lowercase letter in the portable character set,  
 1280 or <underscore>.

1281 to:

1282 The notation [0-9] indicates any digit. The notation [A-Z] indicates any uppercase letter in the portable  
 1283 character set. The notation [Xa-z] indicates the character 'X' or any lowercase letter in the portable  
 1284 character set. The notation [0-9A-Za-z\_]\* indicates zero or more occurrences of any of the following: a  
 1285 digit, an uppercase or lowercase letter in the portable character set, or an <underscore>.

1286 *Rationale:* Austin Group Defect Report(s) applied: 790. See <http://austingroupbugs.net/view.php?id=790>.

1287 **Change Number: XSH/TC2/D4/0008** [790]

1288 On Page: 475-476 Line: 16095-16178 Section: 2.2.2 The Name Space

1289 In the table of reserved identifiers:

1290 Change cexmp1 to cexpm1.  
1291 Change cexmp1f to cexpm1f.  
1292 Change cexmp11 to cexpm11.  
1293 Change ltime to ctime.  
1294 Delete isblank and iswblank.  
1295 Delete strtod, strtodmax, strtoll, strtoull, and strtoumax.  
1296 Delete wcstod, wcstodmax, wcstold, wcstoll, wcstoull, and wcstoumax.  
1297 Delete wcwidth.  
1298 Delete wmem[a-z]\*.

1299 Add the following to the table of reserved identifiers in the correct alphabetical position:

1300 btowc  
1301 ctanh  
1302 ctanhf  
1303 ctanhf  
1304 erf  
1305 erfc  
1306 hypot  
1307 lgamma  
1308 lldiv  
1309 math\_errhandling  
1310 nextafter  
1311 remainder  
1312 snprintf  
1313 va\_copy  
1314 vsnprintf  
1315 wmemchr  
1316 wmemcmp  
1317 wmemcpy  
1318 wmemmove  
1319 wmemset

1320 Remove one of the two appearances of each of the following from the table of reserved identifiers:

1321 acosl  
1322 asinl  
1323 atanf  
1324 atanh  
1325 atanl  
1326 catanh  
1327 catanhf  
1328 catanhf  
1329 ceilf  
1330 ceil  
1331 ldiv

1332 On line 16178 add a new paragraph after the table:

1333 <small>**Note:** The notation [a-z] indicates any lowercase letter in the portable character set. The notation \* indicates any sequence of zero or more characters that are valid in identifiers with external linkage.</small>

1336 *Rationale:* Austin Group Defect Report(s) applied: 790. See <http://austingroupbugs.net/view.php?id=790>.

1337 **Change Number: XSH/TC2/D4/0009** [496]

1338 On Page: 482 Line: 16415 Section: 2.3 Error Numbers

1339 Change the description of [ENOSYS] from:

1340 Function not implemented. An attempt was made to use a function that is not available in this implementation.

1342 to:

1343 Functionality not supported. An attempt was made to use optional functionality that is not supported in this implementation.

1345 *Rationale:* Austin Group Defect Report(s) applied: 496. See <http://austingroupbugs.net/view.php?id=496>.

1346 **Change Number: XSH/TC2/D4/0010** [681]

1347 On Page: 482 Line: 16442 Section: 2.3 Error Numbers  
 (2013 edition Page: 486 Line: 16552)

1349 Change from:

1350 EOPNOTSUP

1351 to:

1352 EOPNOTSUPP

1353 *Rationale:* Austin Group Defect Report(s) applied: 681. See <http://austingroupbugs.net/view.php?id=681>.

1354 **Change Number: XSH/TC2/D4/0011** [690]

1355 On Page: 487 Line: 16657 Section: 2.4.3 Signal Actions  
 (2013 edition Page: 491 Line: 16773-16779)

1357 In the description of SIG\_IGN, change from:

1358 If a process sets the action for the SIGCHLD signal to SIG\_IGN, the behavior is unspecified, [XSI]except as specified below.

1360 If the action for the SIGCHLD signal is set to SIG\_IGN, child processes of the calling processes shall not be transformed into zombie processes when they terminate. If the calling process subsequently waits for its

1362 children, and the process has no unwaited-for children that were transformed into zombie processes, it shall  
 1363 block until all of its children terminate, and *wait()*, *waitid()*, and *waitpid()* shall fail and set *errno* to  
 1364 [ECHILD].[/XSI]

1365 to:

1366 If a process sets the action for the SIGCHLD signal to SIG\_IGN, the behavior is unspecified[XSI], except  
 1367 as specified under "Consequences of Process Termination" in the description of the *\_Exit()* function on  
 1368 page 549[/XSI].

1369 *Rationale*: Austin Group Defect Report(s) applied: 690. See <http://austingroupbugs.net/view.php?id=690>.

1370 **Change Number: XSH/TC2/D4/0012** [516]

1371 On Page: 489 Line: 16721 Section: 2.4.3 Signal Concepts

1372 Change from:

1373 Therefore, applications can invoke them, without restriction, from signal-catching functions:

1374 to:

1375 Therefore, applications can call them, without restriction, from signal-catching functions. Note that,  
 1376 although there is no restriction on the calls themselves, for certain functions there are restrictions on  
 1377 subsequent behavior after the function is called from a signal-catching function (see [xref to *longjmp()*]).

1378 At line 16722-16755 add *longjmp()* and *siglongjmp()* to the table of async-signal-safe functions.

1379 *Rationale*: Austin Group Defect Report(s) applied: 516. See <http://austingroupbugs.net/view.php?id=516>.

1380 **Change Number: XSH/TC2/D4/0013** [692]

1381 On Page: 489 Line: 16722 Section: 2.4.3 Signal Actions  
 1382 (2013 edition Page: 494 Line: 16845-16879)

1383 In the table of async-signal-safe functions, add the following in proper sorted order:

1384 *ffs()*, *htonl()*, *hton()*, *memccpy()*, *memchr()*, *memcmp()*, *memcpy()*, *memmove()*, *memset()*, *ntohl()*, *ntohs()*,  
 1385 *stpcpy()*, *stpncpy()*, *strcat()*, *strchr()*, *strcmp()*, *strcpy()*, *strcspn()*, *strlen()*, *strncat()*, *strncpy()*,  
 1386 *strnlen()*, *strpbrk()*, *strrchr()*, *strspn()*, *strstr()*, *strtok\_r()*, *wcpcpy()*, *wcpncpy()*, *wcsat()*, *wcschr()*,  
 1387 *wcscmp()*, *wcscpy()*, *wcscspn()*, *wcslen()*, *wcsncat()*, *wcsncmp()*, *wcsncpy()*, *wcsnlen()*, *wcspbrk()*,  
 1388 *wcschr()*, *wcsspn()*, *wcsstr()*, *wcstok()*, *wmemchr()*, *wmemcmp()*, *wmemmove()*, *wmemset()*

1389 *Rationale*: Austin Group Defect Report(s) applied: 692. See <http://austingroupbugs.net/view.php?id=692>.  
 1390 Since these functions do not modify or depend on global state, there are no known implementations where  
 1391 these functions cannot be safely used in aysnc-signal-safe code.

1392 **Change Number: XSH/TC2/D4/0014** [615]

1393 On Page: 489 Line: 16756 Section: 2.4.3 Signal Concepts

1394 Change from:

1395 All functions not in the above table are considered to be unsafe with respect to signals.

1396 to:

1397 Any function not in the above table may be unsafe with respect to signals. Implementations may make  
 1398 other interfaces *async-signal-safe*.

1399 *Rationale*: Austin Group Defect Report(s) applied: 615. See <http://austingroupbugs.net/view.php?id=615>.

1400 **Change Number: XSH/TC2/D4/0015** [516]

1401 On Page: 490 Line: 16758 Section: 2.4.3 Signal Concepts

1402 Change from:

1403 ... with a single exception: when a signal interrupts an unsafe function and the signal-catching function calls  
 1404 an unsafe function, the behavior is undefined.

1405 to:

1406 ... with the exception that when a signal interrupts an unsafe function or equivalent (such as the processing  
 1407 equivalent to *exit()* performed after a return from the initial call to *main()*) and the signal-catching function  
 1408 calls an unsafe function, the behavior is undefined. Additional exceptions are specified in the descriptions  
 1409 of individual functions such as *longjmp()*.

1410 *Rationale*: Austin Group Defect Report(s) applied: 516. See <http://austingroupbugs.net/view.php?id=516>.  
 1411 The restrictions on using *longjmp()* and *siglongjmp()* are more restrictive than they need to be on POSIX  
 1412 systems. The loosened restrictions presented here do not break existing implementations and make it easier  
 1413 for application writers to create portable applications.

1414 **Change Number: XSH/TC2/D4/0016** [807]

1415 On Page: 490 Line: 16761 Section: 2.4.3 Signal Concepts

1416 Change from:

1417 Operations which obtain the value of *errno* and operations which assign a value to *errno* shall be *async-signal-safe*.

1419 to:

1420 Operations which obtain the value of *errno* and operations which assign a value to *errno* shall be *async-signal-safe*, provided that the signal-catching function saves the value of *errno* upon entry and restores it  
 1421 before it returns.

1423 *Rationale*: Austin Group Defect Report(s) applied: 807. See <http://austingroupbugs.net/view.php?id=807>.

1424

1425 **Change Number: XSH/TC2/D4/0017** [807]

1426 On Page: 490 Line: 16775 Section: 2.4.4 Signal Effects on Other Functions  
 1427 (2013 edition Page: 495 Line: 16899)

1428 Change from:

1429 ... except as noted for unsafe functions.

1430 to:

1431 ... except as noted for unsafe functions. After returning from a signal-catching function, the value of *errno*  
 1432 is unspecified if the signal-catching function or any function it called assigned a value to *errno* and the  
 1433 signal-catching function did not save and restore the original value of *errno*.

1434 *Rationale*: Austin Group Defect Report(s) applied: 807. See <http://austingroupbugs.net/view.php?id=807>.

1435 **Change Number: XSH/TC2/D4/0018** [608]

1436 On Page: 490 Line: 16788 Section: 2.5 Standard I/O Streams

1437 Change from:

1438 All input takes place as if bytes were read by successive calls to *fgetc()*; all output takes place as if bytes  
 1439 were written by successive calls to *fputc()*.

1440 to:

1441 The wide character input functions shall read characters from the stream and convert them to wide  
 1442 characters as if they were read by successive calls to the *fgetwc()* function. Each conversion shall occur as  
 1443 if by a call to the *mbrtowc()* function, with the conversion state described by the stream's own **mbstate\_t**  
 1444 object (see [xref to 2.5.2]). The byte input functions shall read characters from the stream as if by  
 1445 successive calls to the *fgetc()* function.

1446 The wide character output functions shall convert wide characters to characters and write them to the  
 1447 stream as if they were written by successive calls to the *fputwc()* function. Each conversion shall occur as if  
 1448 by a call to the *wcrtomb()* function, with the conversion state described by the stream's own **mbstate\_t**  
 1449 object (see [xref to 2.5.2]). The byte output functions shall write characters to the stream as if by successive  
 1450 calls to the *fputc()* function.

1451 The *perror()*, *psiginfo()* and *psignal()* functions shall behave as described above for the byte output  
 1452 functions if the stream is already byte-oriented, and shall behave as described above for the wide character  
 1453 output functions if the stream is already wide-oriented. If the stream has no orientation, they shall behave as  
 1454 described for the byte output functions except that they shall not change the orientation of the stream.

1455 Functions other than *perror()*, *psiginfo()* and *psignal()* that write to streams but are neither wide character  
 1456 output nor byte output functions ( *getopt()* and *wordexp()*), shall behave as described above for the byte  
 1457 output functions, except that if the stream has no orientation, it is unspecified whether they set the stream to  
 1458 byte orientation or leave it with no orientation.

1459 *Rationale*: Austin Group Defect Report(s) applied: 608. See <http://austingroupbugs.net/view.php?id=608>.

1460 **Change Number: XSH/TC2/D4/0019** [480]

1461 On Page: 491 Line: 16844 Section: 2.5.1 Interaction of File Descriptors and Standard I/O Streams

1462 Change from:

1463 A handle which is a stream is considered to be closed when either an *fclose()* or *freopen()* is executed on it  
 1464 (the result of *freopen()* is a new stream, which cannot be a handle on the same open file description as its  
 1465 previous value),

1466 to:

1467 A handle which is a stream is considered to be closed when either an *fclose()*, or *freopen()* with non-null  
 1468 filename, is executed on it (for *freopen()* with a null filename, it is implementation-defined whether a new  
 1469 handle is created or the existing one reused),

1470 *Rationale:* Austin Group Defect Report(s) applied: 480. See <http://austingroupbugs.net/view.php?id=480>.  
 1471 The standard is contradictory on whether *freopen()* can reuse a file description when passed a null filename.

1472 **Change Number: XSH/TC2/D4/0020** [631]

1473 On Page: 507 Line: 17490-17492 Section: 2.9.1 Thread-Safety

1474 Remove *getc\_unlocked()*, *getchar\_unlocked()*, *putc\_unlocked()*, and *putchar\_unlocked()* from the list of  
 1475 functions that need not be thread-safe.

1476 *Rationale:* Austin Group Defect Report(s) applied: 631. See <http://austingroupbugs.net/view.php?id=631>.

1477 **Change Number: XSH/TC2/D4/0021** [826]

1478 On Page: 507 Line: 17498 Section: 2.9.1 Thread-Safety

1479 (2013 edition Page: 512 Line: 17626)

1480 Add *setlocale()* to the list of functions that need not be thread-safe.

1481 *Rationale:* Austin Group Defect Report(s) applied: 826. See <http://austingroupbugs.net/view.php?id=826>.

1482 **Change Number: XSH/TC2/D4/0022** [631]

1483 On Page: 507 Line: 17512 Section: 2.9.1 Thread-Safety

1484 Change from:

1485 The *wcrtomb()* and *wcsrtombs()* functions need not be thread-safe if passed a NULL *ps* argument.

1486 to:

1487 The *wcrtomb()* and *wcsrtombs()* functions need not be thread-safe if passed a NULL *ps* argument. The  
 1488 *getc\_unlocked()*, *getchar\_unlocked()*, *putc\_unlocked()*, and *putchar\_unlocked()* functions need not be  
 1489 thread-safe unless the invoking thread owns the (**FILE \***) object accessed by the call, as is the case after a  
 1490 successful call to the *flockfile()* or *ftrylockfile()* functions.

1491 *Rationale:* Austin Group Defect Report(s) applied: 631. See <http://austingroupbugs.net/view.php?id=631>.

1492 **Change Number: XSH/TC2/D4/0023** [627]

1493 On Page: 512 Line: 17704 Section: 2.9.5.2 Cancellation Points

1494 Remove system() from the list of mandatory cancellation points.

1495 *Rationale:* Austin Group Defect Report(s) applied: 627. See <http://austingroupbugs.net/view.php?id=627>.  
 1496 The system() function is not required to be thread-safe, yet the standard requires a cancellation point to  
 1497 occur when executing it.

1498 **Change Number: XSH/TC2/D4/0024** [627,632]

1499 On Page: 513-514 Line: 17715-17789 Section: 2.9.5.2 Cancellation Points

1500 Remove the following functions (which are not required to be thread-safe) from the list of optional  
 1501 cancellation points:

asctime()	endutxent()	getopt()	getutxid()
catgets()	ftw()	getprotobyname()	getutxline()
ctime()	getdate()	getprotobynumber()	localtime()
dbm_close()	getgrent()	getprotoent()	nftw()
dbm_delete()	getgrgid()	getpwent()	pututxline()
dbm_fetch()	getgrnam()	getpwnam()	readdir()
dbm_nextkey()	gethostent()	getpwuid()	setgrent()
dbm_open()	getlogin()	getservbyname()	setpwent()
dbm_store()	getnetbyaddr()	getservbyport()	setutxent()
endgrent()	getnetbyname()	getservent()	strerror()
endpwent()	getnetent()	getutxent()	ttynname()

1502 Remove pclose() from the list of optional cancellation points.

1503 *Rationale:* Austin Group Defect Report(s) applied: 627,632. See  
 1504 <http://austingroupbugs.net/view.php?id=627>.  
 1505 See <http://austingroupbugs.net/view.php?id=632>.

1506

1507 **Change Number: XSH/TC2/D4/0025** [627]

1508 On Page: 514 Line: 17789 Section: 2.9.5.2 Cancellation Points

1509 Add a new paragraph after the list of optional cancellation points:

1510 In addition, a cancellation point may occur when a thread is executing any function that this standard does not require to be thread-safe but the implementation documents as being thread-safe. If a thread is cancelled while executing a non-thread-safe function, the behavior is undefined.

1513 *Rationale:* Austin Group Defect Report(s) applied: 627. See <http://austingroupbugs.net/view.php?id=627>.

1514 **Change Number: XSH/TC2/D4/0026** [632]

1515 On Page: 514 Line: 17795 Section: 2.9.5.2 Cancellation Points

1516 Change from:

1517 Any such side-effects occur before any cancellation cleanup handlers are called.

1518 to:

1519 Any such side-effects occur before any cancellation cleanup handlers are called. For functions that are explicitly required not to return when interrupted (for example, *pclose()*), if a thread is canceled while executing the function, the behavior is undefined.

1522 *Rationale:* Austin Group Defect Report(s) applied: 632. See <http://austingroupbugs.net/view.php?id=632>.

1523 **Change Number: XSH/TC2/D4/0027** [622]

1524 On Page: 515 Line: 17840 Section: 2.9.5.4 Async-Cancel Safety

1525 Add two paragraphs to the end of the section:

1526 If a thread has asynchronous cancellation enabled and is cancelled during execution of a function that is not async-cancel-safe, the behavior is undefined.

1528 If a thread has deferred cancellation enabled, a signal catching function is called in that thread during execution of a function that is not async-cancel-safe, and the signal catching function calls any function that is a cancellation point while a cancellation is pending for the thread, the behavior is undefined.

1531 *Rationale:* Austin Group Defect Report(s) applied: 622. See <http://austingroupbugs.net/view.php?id=622>.

1532 **Change Number: XSH/TC2/D4/0028** [498]

1533 On Page: 516 Line: 17869 Section: 2.9.7 Thread Interactions with Regular File Operations

1534 Append the following text to the last paragraph:

1535 The requirement on the *close()* function shall also apply whenever a file descriptor is successfully closed however caused (for example, as a consequence of calling *close()*, calling *dup2()*, or of process

1537 termination).

1538 *Rationale*: Austin Group Defect Report(s) applied: 498. See <http://austingroupbugs.net/view.php?id=498>.

1539 **Change Number: XSH/TC2/D4/0029** [972]

1540 On Page: 517 Line: 17899 Section: 2.9 Threads  
 1541 (2013 edition Page: 522 Line: 18031)

1542 Add a new section:

### 1543 2.9.9 Synchronization Object Copies and Alternative Mappings

1544 For barriers, condition variables, mutexes, and read-write locks, [TSH]if the *process-shared* attribute is set  
 1545 to PTHREAD\_PROCESS\_PRIVATE,[/TSH] only the synchronization object at the address used to  
 1546 initialize it can be used for performing synchronization. The effect of referring to another mapping of the  
 1547 same object when locking, unlocking, or destroying the object is undefined. [TSH]If the *process-shared*  
 1548 attribute is set to PTHREAD\_PROCESS\_SHARED, only the synchronization object itself can be used for  
 1549 performing synchronization; however, it need not be referenced at the address used to initialize it (that is,  
 1550 another mapping of the same object can be used).[/TSH] The effect of referring to a copy of the object  
 1551 when locking, unlocking, or destroying it is undefined.

1552 For spin locks, the above requirements shall apply as if spin locks have a *process-shared* attribute that is set  
 1553 from the *pshared* argument to *pthread\_spin\_init()*. For semaphores, the above requirements shall apply as  
 1554 if semaphores have a *process-shared* attribute that is set to PTHREAD\_PROCESS\_PRIVATE if the  
 1555 *pshared* argument to *sem\_init()* is zero and set to PTHREAD\_PROCESS\_SHARED if *pshared* is non-zero.

1556 *Rationale*: Austin Group Defect Report(s) applied: 972. See <http://austingroupbugs.net/view.php?id=972>.  
 1557 Most implementations of these private barriers, condition variables, mutexes, reader-writer locks,  
 1558 semaphores and spin locks use the address of the object to locate the object when using  
 1559 PTHREAD\_PROCESS\_PRIVATE (or equivalent *pshared* argument). The optimizations that can be  
 1560 performed when an object is not shared with other processes disappear if multiple mappings of one of these  
 1561 objects are allowed in this case. (If a process needs multiple mappings of an object, it can use  
 1562 PTHREAD\_PROCESS\_SHARED instead of PTHREAD\_PROCESS\_PRIVATE to get the desired  
 1563 behavior.)

1564 **Change Number: XSH/TC2/D4/0030** [733]

1565 On Page: 541 Line: 18857 Section: 2.12.1 Defined Types  
 1566 (2013 edition Page: 546 Line: 18989)

1567 In the description of *sig\_atomic\_t*, change from:

1568 Integer type ...

1569 to:

1570 Possibly volatile-qualified integer type ...

1571 *Rationale*: Austin Group Defect Report(s) applied: 733. See <http://austingroupbugs.net/view.php?id=733>.

1572

1573 **Change Number: XSH/TC2/D4/0031** [690]

1574 On Page: 541 Line: 18886 Section: 2 General Information  
 1575 (2013 edition Page: 546 Line: 19012)

1576 Add new section:

1577 **2.13 Status Information**

1578 Status information is data associated with a process detailing a change in the state of the process. It shall  
 1579 consist of:

- 1580 • The state the process transitioned into ('stopped', 'continued', or 'terminated').
- 1581 • The information necessary to populate the **siginfo\_t** structure provided by *waitid()*.
- 1582 • If the new state is 'terminated':
  - 1583 — The low-order 8 bits of the status argument that the process passed to *\_Exit()*, *\_exit()*, or  
   1584 *exit()*, or the low-order 8 bits of the value the process returned from *main()*. Note that these 8  
   1585 bits are part of the complete value that is used to set the *si\_status* member of the **siginfo\_t**  
   1586 structure provided by *waitid()*.
  - 1587 — Whether the process terminated due to the receipt of a signal that was not caught, and if so,  
   1588 the number of the signal that caused the termination of the process.
- 1589 • If the new state is 'stopped':
  - 1590 — The number of the signal that caused the process to stop.

1591 A process might not have any status information (such as immediately after a process has started).

1592  
 1593 Status information for a process shall be generated (made available to the parent process) when the process  
 1594 stops, continues, or terminates except in the following case:

- 1595 • If the parent process sets the action for the SIGCHLD signal to SIG\_IGN, or if the parent sets the  
   1596 **SA\_NOCLDWAIT** flag for the SIGCHLD signal action, process termination shall not generate  
   1597 new status information but shall cause any existing status information for the process to be  
   1598 discarded.

1599 If new status information is generated, and the process already had status information, the existing status  
 1600 information shall be discarded and replaced with the new status information.

1601  
 1602 Only the process's parent process can obtain the process's status information. The parent obtains a child's  
 1603 status information by calling *wait()*, *waitid()*, or *waitpid()*.

1604  
 1605 Except when *waitid()* is called with the WNOWAIT flag set in the options argument, the status information  
 1606 obtained by a wait function shall be consumed (discarded) by that wait function; no two calls to *wait()*,  
 1607 *waitid()* (without WNOWAIT), or *waitpid()* shall obtain the same status information.

1608  
 1609 When status information becomes available to the parent process and more than one thread in the parent  
 1610 process is waiting for the status information (blocked in a call to *wait()*, *waitid()*, or *waitpid()* with  
 1611 arguments that would match the status information):

- 1612 • If none of the matching threads is in a call to *waitid()* with the WNOWAIT flag set in the options  
   1613 argument, the thread that obtains the status information is unspecified.
- 1614 • Otherwise (at least one of the matching threads is in a call to *waitid()* with the WNOWAIT flag  
   1615 set), the matching thread or threads that obtain the status information is unspecified except that at  
   1616 least one of the matching threads shall obtain the status information and at most one of the

1617 matching threads that are not calling *waitid()* with the WNOWAIT flag set shall obtain it.

1618 *Rationale:* Austin Group Defect Report(s) applied: 690. See <http://austingroupbugs.net/view.php?id=690>.

1619 **Change Number: XSH/TC2/D4/0032** [835]

1620 On Page: 541 Line: 18886 Section: 2 General Information

1621 Add new section:

1622 **2.14 File Descriptor Allocation**

1623 All functions that open one or more file descriptors shall, unless specified otherwise, atomically allocate the  
 1624 lowest numbered available (that is, not already open in the calling process) file descriptor at the time of  
 1625 each allocation. Where a single function allocates two file descriptors (for example *pipe()* or *socketpair()*),  
 1626 the allocations may be independent and therefore applications should not expect them to have adjacent  
 1627 values or depend on which has the higher value.

1628 *Rationale:* Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

1629 **Change Number: XSH/TC2/D4/0033** [594]

1630 On Page: 545 Line: 18913 Section: *\_Exit()*

1631 In the DESCRIPTION section, change from:

1632 the least significant 8 bits (that is, *status* & 0377) shall be available to a waiting parent process

1633 to:

1634 the least significant 8 bits (that is, *status* & 0377) shall be available from *wait()* and *waitpid()*; the full value  
 1635 shall be available from *waitid()* and in the **siginfo\_t** passed to a signal handler for SIGCHLD.

1636 *Rationale:* Austin Group Defect Report(s) applied: 594. See <http://austingroupbugs.net/view.php?id=594>.

1637 **Change Number: XSH/TC2/D4/0034** [594,690]

1638 On Page: 545 Line: 18925 Section: *\_Exit()*

1639 (2013 edition Page: 549 Line: 19051)

1640 At "Consequences of Process Termination" change from:

1641 • If the parent process of the calling process is executing a *wait()*, *waitid()*, or *waitpid()*, [XSI] and  
 1642 has neither set its SA\_NOCLDWAIT flag nor set SIGCHLD to SIG\_IGN,[/XSI] it shall be  
 1644 notified of termination of the calling process and the low-order eight bits (that is, *bits* 0377) of  
 1645 *status* shall be made available to it. If the parent is not waiting, the child's status shall be made  
 1645 available to it when the parent subsequently executes *wait()*, *waitid()*, or *waitpid()*.

1646 The semantics of the *waitid()* function shall be equivalent to *wait()*.

1647 • If the parent process of the calling process is not executing a *wait()*, *waitid()*, or *waitpid()*,

1648 [XSI]and has neither set its SA\_NOCLDWAIT flag nor set SIGCHLD to SIG\_IGN,[/XSI] the  
 1649 calling process shall be transformed into a zombie process. A zombie process is an inactive  
 1650 process and it shall be deleted at some later time when its parent process executes *wait()*, *waitid()*,  
 1651 or *waitpid()*.

1652 [XSI]The semantics of the *waitid()* function shall be equivalent to *wait()*.[/XSI]

- 1653 • Termination of a process does not directly terminate its children. The sending of a SIGHUP signal  
 1654 as described below indirectly terminates children in some circumstances.

- 1655 • Either:

1656     1657     If the implementation supports the SIGCHLD signal, a SIGCHLD shall be sent to the parent  
 1658     process.

1659     1660     Or:

1661     1662     [XSI]If the parent process has set its SA\_NOCLDWAIT flag, or set SIGCHLD to SIG\_IGN, the  
 1663     status shall be discarded, and the lifetime of the calling process shall end immediately. If  
 1664     SA\_NOCLDWAIT is set, it is implementation-defined whether a SIGCHLD signal is sent to the  
 1665     parent process.[/XSI]

1666     to:

- 1667     • [XSI]If the parent process of the calling process has set its SA\_NOCLDWAIT flag or has set the  
 1668     action for the SIGCHLD signal to SIG\_IGN:
  - 1669         — The process's status information (see XSH Section 2.13), if any, shall be discarded.
  - 1670         — The lifetime of the calling process shall end immediately. If SA\_NOCLDWAIT is set, it is  
 1671             implementation-defined whether a SIGCHLD signal is sent to the parent process.
  - 1672         — If a thread in the parent process of the calling process is blocked in *wait()*, *waitpid()*, or  
 1673             *waitid()*, and the parent process has no remaining child processes in the set of waited-for  
 1674             children, the *wait()*, *waitid()*, or *waitpid()* function shall fail and set *errno* to [ECHILD].
- 1675     Otherwise:[/XSI]
  - 1676         — Status information (see XSH Section 2.13) shall be generated.
  - 1677         — The calling process shall be transformed into a zombie process. Its status information shall be  
 1678             made available to the parent process until the process's lifetime ends.
  - 1679         — The process's lifetime shall end once its parent obtains the process's status information via a  
 1680             currently-blocked or future call to *wait()*, *waitid()* (without WNOWAIT), or *waitpid()*.
  - 1681         — If one or more threads in the parent process of the calling process is blocked in a call to  
 1682             *wait()*, *waitid()*, or *waitpid()* awaiting termination of the process, one (or, if any are calling  
 1683             *waitid()* with WNOWAIT, possibly more) of these threads shall obtain the process's status  
 1684             information as specified in XSH Section 2.13 and become unblocked.
  - 1685         — A SIGCHLD shall be sent to the parent process.
- 1686     • Termination of a process does not directly terminate its children. The sending of a SIGHUP signal  
 1687     as described below indirectly terminates children in some circumstances.

1688     1689     Rationale: Austin Group Defect Report(s) applied: 594,690. See  
 1690     <http://austingroupbugs.net/view.php?id=594>.  
 1690     See <http://austingroupbugs.net/view.php?id=690>.

1691 **Change Number: XSH/TC2/D4/0035** [835]

1692 On Page: 559 Line: 19374 Section: accept()  
1693 (2013 edition Page: 563 Line: 19500)

1694 After the text:

1695 ... and allocate a new file descriptor for that socket.

1696 add:

1697 The file descriptor shall be allocated as described in [xref to new section 2.14].

1698 *Rationale:* Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

1699 **Change Number: XSH/TC2/D4/0036** [836]

1700 On Page: 559 Line: 19398 Section: accept()

1701 In the RETURN VALUE section, change from:

1702 Otherwise, -1 shall be returned and *errno* set to indicate the error.

1703 to:

1704 Otherwise, -1 shall be returned, *errno* shall be set to indicate the error, and any object pointed to by  
1705 *address\_len* shall remain unchanged.

1706 *Rationale:* Austin Group Defect Report(s) applied: 836. See <http://austingroupbugs.net/view.php?id=836>.

1707 **Change Number: XSH/TC2/D4/0037** [873]

1708 On Page: 561 Line: 19442 Section: access()  
1709 (2013 edition Page: 565 Line: 19569)

1710 In the NAME section, delete:

1711 relative to directory file descriptor

1712 *Rationale:* Austin Group Defect Report(s) applied: 873. See <http://austingroupbugs.net/view.php?id=873>.

1713 **Change Number: XSH/TC2/D4/0038** [591]

1714 On Page: 561 Line: 19444 Section: access()

1715 Before the faccessat SYNOPSIS line, insert a line with OH shading:

1716 `#include <fcntl.h>`

1717 *Rationale:* Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

1718 **Change Number: XSH/TC2/D4/0039** [838]

1719 On Page: 561 Line: 19449 Section: access()  
 1720 (2013 edition Page: 565 Line: 19576)

1721 In the DESCRIPTION section, change from:

1722 ... for accessibility according to the bit pattern contained in *amode*, using the real user ID in place of the  
 1723 effective user ID and the real group ID in place of the effective group ID.

1724 to:

1725 ... for accessibility according to the bit pattern contained in *amode*. The checks for accessibility (including  
 1726 directory permissions checked during pathname resolution) shall be performed using the real user ID in  
 1727 place of the effective user ID and the real group ID in place of the effective group ID.

1728 On Page: 561 Line: 19457 Section: access()  
 1729 (2013 edition Page: 565 Line: 19584)

1730 In the DESCRIPTION section, change from:

1731 The *faccessat()* function shall be equivalent to the *access()* function, except ...

1732 to:

1733 The *faccessat()* function when called with a *flag* value of zero shall be equivalent to the *access()* function,  
 1734 except ...

1735 *Rationale:* Austin Group Defect Report(s) applied: 838. See <http://austingroupbugs.net/view.php?id=838>.

1736 **Change Number: XSH/TC2/D4/0040** [817]

1737 On Page: 561 Line: 19460 Section: access()  
 1738 (2013 edition Page: 565 Line: 19587)

1739 In the DESCRIPTION section, change from:

1740 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
 1741 are permitted using the current permissions of the directory underlying the file descriptor. If the file  
 1742 descriptor was opened with O\_SEARCH, the function shall not perform the check.

1743 to:

1744 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
 1745 function shall check whether directory searches are permitted using the current permissions of the directory  
 1746 underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

1747 *Rationale:* Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

1748

1749 **Change Number: XSH/TC2/D4/0041** [487]

1750 On Page: 561 Line: 19464 Section: access()

1751 In the DESCRIPTION section, change from:

1752 If *faccessat()* is passed the special value AT\_FDCWD in the *fd* parameter, the current working directory is  
1753 used and the behavior shall be identical to a call to *access()*.

1754 to:

1755 If *faccessat()* is passed the special value AT\_FDCWD in the *fd* parameter, the current working directory  
1756 shall be used and, if *flag* is zero, the behavior shall be identical to a call to *access()*.

1757 *Rationale:* Austin Group Defect Report(s) applied: 487. See <http://austingroupbugs.net/view.php?id=487>.

1758 **Change Number: XSH/TC2/D4/0042** [838]

1759 On Page: 561 Line: 19468 Section: access()  
1760 (2013 edition Page: 565 Line: 19595)

1761 In the DESCRIPTION section, for AT\_EACCESS change from:

1762 The checks for accessibility are performed using the effective user and group IDs instead of the real user  
1763 and group ID

1764 to:

1765 The checks for accessibility (including directory permissions checked during pathname resolution) shall be  
1766 performed using the effective user ID and group ID instead of the real user ID and group ID

1767 *Rationale:* Austin Group Defect Report(s) applied: 838. See <http://austingroupbugs.net/view.php?id=838>.

1768 **Change Number: XSH/TC2/D4/0043** [817]

1769 On Page: 562 Line: 19488 Section: access()  
1770 (2013 edition Page: 566 Line: 19616)

1771 In the ERRORS section, for the [EACCES] error, change from:

1772 *fd* was not opened with O\_SEARCH and ...

1773 to:

1774 The access mode of the open file description associated with *fd* is not O\_SEARCH and ...

1775 *Rationale:* Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

1776

1777 **Change Number: XSH/TC2/D4/0044** [838]

1778 On Page: 562 Line: 19515 Section: access()  
 1779 (2013 edition Page: 566 Line: 19643)

1780 In the APPLICATION USAGE section, add the following paragraphs to the beginning of the section:

1781 Use of these functions is discouraged since by the time the returned information is acted upon, it is out of  
 1782 date. (That is, acting upon the information always leads to a time-of-check-to-time-of-use race condition.)  
 1783 An application should instead attempt the action itself and handle the EACCES error that occurs if the file  
 1784 is not accessible (with a change of effective user and group IDs beforehand, and perhaps a change back  
 1785 afterwards, in the case where *access()* or *faccessat()* without AT\_EACCES would have been used.)

1786 Historically, one of the uses of *access()* was in set-user-ID root programs to check whether the user running  
 1787 the program had access to a file. This relied on "super-user" privileges which were granted based on the  
 1788 effective user ID being zero, so that when *access()* used the real user ID to check accessibility those  
 1789 privileges were not taken into account. On newer systems where privileges can be assigned which have no  
 1790 association with user or group IDs, if a program with such privileges calls *access()*, the change of IDs has  
 1791 no effect on the privileges and therefore they are taken into account in the accessibility checks. Thus  
 1792 *access()* (and *faccessat()* with *flag* zero) cannot be used for this historical purpose in such programs.  
 1793 Likewise, if a system provides any additional or alternate file access control mechanisms that are not user  
 1794 ID based, they will still be taken into account.

1795 If a relative pathname is used, no account is taken of whether the current directory (or the directory  
 1796 associated with the file descriptor *fd*) is accessible via any absolute pathname. Applications using *access()*,  
 1797 or *faccessat()* without AT\_EACCES, may consequently act as if the file would be accessible to a user with  
 1798 the real user ID and group ID of the process when such a user would not in practice be able to access the  
 1799 file because access would be denied at some point above the current directory (or the directory associated  
 1800 with the file descriptor *fd*) in the file hierarchy.

1801 If *access()* or *faccessat()* is used with W\_OK to check for write access to a directory which has the  
 1802 S\_ISVTX bit set, a return value indicating the directory is writable can be misleading since some  
 1803 operations on files in the directory would not be permitted based on the ownership of those files (see [xref  
 1804 to XBD 4.2]).

1805 On Page: 563 Line: 19544 Section: access()  
 1806 (2013 edition Page: 567 Line: 19672)

1807 Change the FUTURE DIRECTIONS section from:

1808 None.

1809 to:

1810 These functions may be formally deprecated (for example by shading them OB) in a future revision of this  
 1811 standard.

1812 *Rationale:* Austin Group Defect Report(s) applied: 838. See <http://austingroupbugs.net/view.php?id=838>.

1813

1814   **Change Number: XSH/TC2/D4/0045** [671]

1815   On Page: 573 Line: 19824 Section: aio\_fsync()

1816   In the ERRORS section, change from:

1817   The *aio\_fildes* member of the **aiocb** structure referenced by the *aiocbp* argument is not a valid file  
1818   descriptor open for writing.

1819   to:

1820   The *aio\_fildes* member of the **aiocb** structure referenced by the *aiocbp* argument is not a valid file  
1821   descriptor.

1822   On Page: 574 Line: 19973 Section: aio\_fsync()

1823   In the APPLICATION USAGE section, change from:

1824   None.

1825   to:

1826   Note that even if the file descriptor is not open for writing, if there are any pending write requests on the  
1827   underlying file, then that I/O will be completed prior to the return of a call to *aio\_error()* or *aio\_return()*  
1828   indicating that the operation has completed.

1829   *Rationale:* Austin Group Defect Report(s) applied. 671. See <http://austingroupbugs.net/view.php?id=671>.  
1830   The access mode of the file descriptor does not affect whether there are pending I/O operations on the  
1831   underlying file.

1832   **Change Number: XSH/TC2/D4/0046** [892]

1833   On Page: 613 Line: 20931 Section: atol()  
1834   (2013 edition Page: 618 Line: 21108)

1835   In the SYNOPSIS section, change from:

1836   long atol(const char \*str);

1837   to:

1838   long atol(const char \*nptr);

1839   On Page: 613 Line: 20937 Section: atol()  
1840   (2013 edition Page: 618 Line: 21114)

1841   In the DESCRIPTION section, change from:

1842   The call *atol(str)* shall be equivalent to:

1843    `strtol(str, (char **)NULL, 10)`

1844 to:

1845 Except as noted below, the call *atol(nptr)* shall be equivalent to:

1846 `strtol(nptr, (char **)NULL, 10)`

1847 On Page: 613 Line: 20939 section *atol()*  
 1848 (2013 edition Page: 618 Line: 21116)

1849 In the DESCRIPTION section, change from:

1850 The call to *atoll(nptr)* shall be equivalent ...

1851 to:

1852 Except as noted below, the call to *atoll(nptr)* shall be equivalent ...

1853 On Page: 613 Line: 20941 section *atol()*  
 1854 (2013 edition Page: 618 Line: 21118)

1855 In the DESCRIPTION section, change from:

1856 except that the handling of errors may differ.

1857 to:

1858 The handling of errors may differ.

1859 On Page 613 Line: 20950 section *atol()*  
 1860 (2013 edition Page: 618 Line: 21127)

1861 In the APPLICATION USAGE section, change from:

1862 The *atol()* function is subsumed by *strtol()* but is retained because it is used extensively in existing code. If  
 1863 the number is not known to be in range, *strtol()* should be used because *atol()* is not required to perform  
 1864 any error checking.

1865 to:

1866 If the number is not known to be in range, *strtol()* or *strtoll()* should be used because *atol()* and *atoll()* are  
 1867 not required to perform any error checking.

1868 *Rationale:* Austin Group Defect Report(s) applied: 892. See <http://austingroupbugs.net/view.php?id=892>.

1869 **Change Number: XSH/TC2/D4/0047** [656]

1870 On Page: 614 Line: 20979 Section: *basename()*

1871 In the DESCRIPTION section, change from:

1872 The *basename()* function may modify the string pointed to by *path*, and may return a pointer to static

1873 storage that may then be overwritten by a subsequent call to *basename*().

1874 to:

1875 The *basename*() function may modify the string pointed to by *path*, and may return a pointer to internal  
 1876 storage. The returned pointer might be invalidated or the storage might be overwritten by a subsequent call  
 1877 to *basename*(). The returned pointer might also be invalidated if the calling thread is terminated.

1878 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 1879 This item is a layered change on XSH/TC1/D5/0041 [75].

1880 The change is to add the following text to the end of the paragraph at 2013 edition P619, L21156-21158:

1881 The returned pointer might also be invalidated if the calling thread is terminated.

1882 **Change Number: XSH/TC2/D4/0048** [928]

1883 On Page: 614 Line: 20992 Section: *basename*()  
 1884 (2013 edition Page: 619 Line: 21170)

1885 In the EXAMPLES section, change from:

1886 `char *name = "/usr/lib";`

1887 to:

1888 `char name[ ] = "/usr/lib";`

1889 *Rationale*: Austin Group Defect Report(s) applied: 928. See <http://austingroupbugs.net/view.php?id=928>.

1890 **Change Number: XSH/TC2/D4/0049** [612]

1891 On Page: 614 Line: 20996-21004 Section: *basename*()

1892 In the EXAMPLES section change from:

1893 **Sample Input and Output Strings for basename()**

1894  
 1895 In the following table, the input string is the value pointed to by *path*, and the output string is the return  
 1896 value of the *basename*() function.

Input String	Output String
<code>"/usr/lib"</code>	<code>"lib"</code>
<code>"/usr/"</code>	<code>"usr"</code>
<code>" / "</code>	<code>" / "</code>

" / / / "	" / "
"//usr//lib//"	"lib"

1897

1898 to:

1899 **Sample Input and Output Strings for the *basename()* and *dirname()* functions and the *basename* and  
 1900 *dirname* utilities**

<i>basename()</i> and <i>dirname()</i> <i>functions path</i> <i>argument</i>	<i>string returned</i> <i>by basename()</i>	<i>string</i> <i>returned by</i> <i>dirname()</i>	<i>basename</i> and <i>dirname</i> utilities <i>string operand</i>	<i>output written</i> <i>by basename</i> <i>utility</i>	<i>output</i> <i>written by</i> <i>dirname</i> <i>utility</i>
"usr"	"usr"	". "	usr	usr	.
"usr/ "	"usr"	". "	usr/	usr	.
" "	". "	". "	" "	. or empty string	.
" / "	" / "	" / "	/	/	/
" // "	" / " or " // "	" / " or " // "	/	/ or //	/ or //
" // / "	" / "	" / "	///	/	/
" /usr/ "	"usr"	"X"	/usr/	usr	/
" /usr/lib"	"lib"	"/usr"	/usr/lib	lib	/usr
"//usr//lib//"	"lib"	"/usr"	//usr//lib//	lib	//usr
" /home//dwc//te st"	"test"	"/home//dwc "	/home//dwc//te st	test	/home//dwc

1901

1902 *Rationale:* Austin Group Defect Report(s) applied: 612. See <http://austingroupbugs.net/view.php?id=612>.1903 **Change Number: XSH/TC2/D4/0050** [822]

1904 On Page: 617 Line: 21086 Section: bind()  
 1905 (2013 edition Page: 622 Line: 21267)

1906 In the ERRORS section, change from:

1907 [ENOENT]

1908 A component of the pathname does not name an existing file or the pathname is an empty string.

1909 to:

1910 [ENOENT]

1911 A component of the path prefix of the pathname in *address* does not name an existing file or the  
 1912 pathname is an empty string.

1913 [ENOENT] or [ENOTDIR]

1914 The pathname in *address* contains at least one non-<slash> character and ends with one or more  
 1915 trailing <slash> characters. If the pathname without the trailing <slash> characters would name an  
 1916 existing file, an [ENOENT] error shall not occur.

1917 *Rationale*: Austin Group Defect Report(s) applied: 822. See <http://austingroupbugs.net/view.php?id=822>.

1918 This change layers upon XSH/TC1/D5/0043 [146], changing the last sentence of the change.

1920 **Change Number: XSH/TC2/D4/0051** [756]

1921 On Page: 620 Line: 21229 Section: bsearch()

1922 In the RATIONALE section, change from:

1923 `((char *)p - (char *)base) % width == 0`

1924 to:

1925 `((char *)p - (char *)base) % width == 0`

1926 *Rationale*: Austin Group Defect Report(s) applied: 756. See <http://austingroupbugs.net/view.php?id=756>.

1927 **Change Number: XSH/TC2/D4/0052** [663]

1928 On Page: 622 Line: 21263 Section: btowc()  
 1929 (2013 edition Page: 627 Line: 21451)

1930 In the RETURN VALUE section, add a new sentence:

1931 [CX]In the POSIX locale, *btowc()* shall not return WEOF if *c* has a value in the range 0 to 255  
 1932 inclusive.[CX]

1933 *Rationale*: Austin Group Defect Report(s) applied: 663. See <http://austingroupbugs.net/view.php?id=663>.

1934 **Change Number: XSH/TC2/D4/0053** [526]

1935 On Page: 627 Line: 21395,21399 Section: calloc()

1936 In the DESCRIPTION section, at line 21395 change from:

1937 If the size of the space requested is 0, the behavior is implementation-defined: the value returned shall be  
 1938 either a null pointer or a unique pointer.

1939 to:

1940 If the size of the space requested is 0, the behavior is implementation-defined: either a null pointer shall be  
 1941 returned, or the behavior shall be as if the size were some nonzero value, except that the behavior is  
 1942 undefined if the returned pointer is used to access an object.

1943 In the RETURN VALUE section, at line 21399 change from:

1944 either a null pointer or a unique pointer that can be successfully passed to *free()* shall be returned.

1945 to:

1946 either:

1947 • a null pointer shall be returned [CX] and *errno* may be set to an implementation-defined  
 1948 value[/CX], or  
 1949 • a pointer to the allocated space shall be returned. The application shall ensure that the pointer is  
 1950 not used to access an object.

1951 *Rationale*: Austin Group Defect Report(s) applied: 526. See <http://austingroupbugs.net/view.php?id=526>.

1952 **Change Number: XSH/TC2/D4/0054** [645]

1953 On Page: 639 Line: 21683 Section: *catopen()*

1954 In the DESCRIPTION section, change from:

1955 complete name

1956 to:

1957 pathname

1958 *Rationale*: Austin Group Defect Report(s) applied: 645. See <http://austingroupbugs.net/view.php?id=645>.

1959 **Change Number: XSH/TC2/D4/0055** [497]

1960 On Page: 639 Line: 21685 Section: *catopen()*

1961 In the DESCRIPTION section, change from:

1962 ... on page 173). If *NLSPATH* exists in the environment ...

1963 to:

1964 ... on page 173); if *NLSPATH* exists in the environment ...

1965 *Rationale*: Austin Group Defect Report(s) applied: 497. See <http://austingroupbugs.net/view.php?id=497>.

1966

1967 **Change Number: XSH/TC2/D4/0056** [497]

1968 On Page: 640 Line: 21734 Section: catopen()

1969 In the APPLICATION USAGE section, add a new paragraph to the end of the section:

1970 To be sure that messages produced by an application running with appropriate privileges cannot be used by  
1971 an attacker setting an unexpected value for *NLSPATH* in the environment to confuse a system  
1972 administrator, such applications should use pathnames containing a '/' to get defined behavior when using  
1973 *catopen()* to open a message catalog.

1974 *Rationale:* Austin Group Defect Report(s) applied: 497. See <http://austingroupbugs.net/view.php?id=497>.

1975 **Change Number: XSH/TC2/D4/0057** [873]

1976 On Page: 655 Line: 22155 Section: chmod()  
1977 (2013 edition Page: 660 Line: 22337)

1978 In the NAME section, delete:

1979 relative to directory file descriptor

1980 *Rationale:* Austin Group Defect Report(s) applied: 873. See <http://austingroupbugs.net/view.php?id=873>.

1981 **Change Number: XSH/TC2/D4/0058** [591]

1982 On Page: 655 Line: 22157 Section: chmod()

1983 Before the fchmodat SYNOPSIS line, insert a line with OH shading:

1984 `#include <fcntl.h>`

1985 *Rationale:* Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

1986 **Change Number: XSH/TC2/D4/0059** [817]

1987 On Page: 655 Line: 22178 Section: chmod()  
1988 (2013 edition Page: 660 Line: 22358)

1989 In the DESCRIPTION section, change from:

1990 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
1991 are permitted using the current permissions of the directory underlying the file descriptor. If the file  
1992 descriptor was opened with O\_SEARCH, the function shall not perform the check.

1993 to:

1994 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
1995 function shall check whether directory searches are permitted using the current permissions of the directory  
1996 underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

1997 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

1998 **Change Number: XSH/TC2/D4/0060** [817]

1999 On Page: 656 Line: 22207 Section: chmod()  
 2000 (2013 edition Page: 661 Line: 22388)

2001 In the ERRORS section, for the [EACCES] error, change from:

2002 *fd* was not opened with O\_SEARCH and ...

2003 to:

2004 The access mode of the open file description associated with *fd* is not O\_SEARCH and ...

2005 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

2006 **Change Number: XSH/TC2/D4/0061** [893]

2007 On Page: 657 Line: 22256 Section: chmod()  
 2008 (2013 edition Page: 662 Line: 22437)

2009 In the EXAMPLES section, change from:

2010 `chmod("home/cnd/mod1", S_IRWXU|S_IRWXG|S_IROTH|S_IWOTH);`  
 2011 `status = stat("home/cnd/mod1", &buffer);`

2012 to:

2013 `chmod("/home/cnd/mod1", S_IRWXU|S_IRWXG|S_IROTH|S_IWOTH);`  
 2014 `status = stat("/home/cnd/mod1", &buffer);`

2015 *Rationale*: Austin Group Defect Report(s) applied: 893. See <http://austingroupbugs.net/view.php?id=893>.

2016 **Change Number: XSH/TC2/D4/0062** [873]

2017 On Page: 659 Line: 22310 Section: chown()  
 2018 (2013 edition Page: 664 Line: 22494)

2019 In the NAME section, delete:

2020 relative to directory file descriptor

2021 *Rationale*: Austin Group Defect Report(s) applied: 873. See <http://austingroupbugs.net/view.php?id=873>.

2022 **Change Number: XSH/TC2/D4/0063** [591]

2023 On Page: 659 Line: 22312 Section: chown()

2024 Before the fchownat SYNOPSIS line, insert a line with OH shading:

2025 #include <fcntl.h>

2026 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

2027 **Change Number: XSH/TC2/D4/0064 [485]**

2028 On Page: 659 Line: 22337 Section: chown()

2029 In the DESCRIPTION section, change from:

2030 If owner or group is specified as **(uid\_t)-1** or **(gid\_t)-1**, respectively, the corresponding ID of the file shall  
 2031 not be changed. If both owner and group are -1, the times need not be updated.

2032 Upon successful completion, *chown()* shall mark for update the last file status change timestamp of the file.

2033 to:

2034 If owner or group is specified as **(uid\_t)-1** or **(gid\_t)-1**, respectively, the corresponding ID of the file shall  
 2035 not be changed.

2036 Upon successful completion, *chown()* shall mark for update the last file status change timestamp of the file,  
 2037 except that if owner is **(uid\_t)-1** and group is **(gid\_t)-1**, the file status change timestamp need not be  
 2038 marked for update.

2039 *Rationale*: Austin Group Defect Report(s) applied: 485. See <http://austingroupbugs.net/view.php?id=485>.

2040 The wording for *chown* when both arguments are -1 was difficult to follow.

2041 **Change Number: XSH/TC2/D4/0065 [817]**

2042 On Page: 659 Line: 22343 Section: chown()  
 2043 (2013 edition Page: 664 Line: 22527)

2044 In the DESCRIPTION section, change from:

2045 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
 2046 are permitted using the current permissions of the directory underlying the file descriptor. If the file  
 2047 descriptor was opened with O\_SEARCH, the function shall not perform the check.

2048 to:

2049 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
 2050 function shall check whether directory searches are permitted using the current permissions of the directory  
 2051 underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

2052 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

2053 **Change Number: XSH/TC2/D4/0066 [817]**

2054 On Page: 660 Line: 22376 Section: chown()  
 2055 (2013 edition Page: 665 Line: 22562)

2056 In the ERRORS section, for the [EACCES] error, change from:

2057 *fd* was not opened with O\_SEARCH and ...

2058 to:

2059 The access mode of the open file description associated with *fd* is not O\_SEARCH and ...

2060 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

2061 **Change Number: XSH/TC2/D4/0067** [686]

2062 On Page: 665 Line: 22541-22552 Section: clock()

2063 In the APPLICATION USAGE section, on line 22541 insert a new first paragraph:

2064 In programming environments where **clock\_t** is a 32-bit integer type and CLOCKS\_PER\_SEC is one  
 2065 million, *clock()* will start failing in less than 36 minutes of processor time for signed **clock\_t**, or 72 minutes  
 2066 for unsigned **clock\_t**. Applications intended to be portable to such environments should use *times()* instead  
 2067 (or *clock\_gettime()* with CLOCK\_PROCESS\_CPUTIME\_ID, if supported).

2068 In the APPLICATION USAGE section, on line 22545 delete:

2069 The value returned by *clock()* may wrap around on some implementations. For example, on a machine with  
 2070 32-bit values for **clock\_t**, it wraps after 2147 seconds or 36 minutes.

2071 In the SEE ALSO section, on line 22552 add *clock\_getres()* and *times()* to the SEE ALSO list.

2072 *Rationale*: Austin Group Defect Report(s) applied: 686. See <http://austingroupbugs.net/view.php?id=686>.

2073 **Change Number: XSH/TC2/D4/0068** [909]

2074 On Page: 671 Line: 22759 Section: *clock\_nanosleep()*  
 2075 (2013 edition Page: 676 Line: 22954)

2076 In the RETURN VALUE section, change from:

2077 If the *rmtp* argument is NULL, the remaining time is not returned.

2078 to:

2079 The *rqtp* and *rmtp* arguments can point to the same object. If the *rmtp* argument is NULL, the remaining  
 2080 time is not returned.

2081 *Rationale*: Austin Group Defect Report(s) applied: 909. See <http://austingroupbugs.net/view.php?id=909>.

2082 **Change Number: XSH/TC2/D4/0069** [555]

2083 On Page: 678 Line: 22962 Section: close()

2084 In the APPLICATION USAGE section, add a new paragraph to the end of the section:

2085    Usage of *close()* on file descriptors STDIN\_FILENO, STDOUT\_FILENO or STDERR\_FILENO should  
 2086    immediately be followed by an operation to reopen these file descriptors. Unexpected behavior will result if  
 2087    any of these file descriptors is left in a closed state (for example, an EBADF error from  *perror()*) or if an  
 2088    unrelated *open()* or similar call later in the application accidentally allocates a file to one of these well-  
 2089    known file descriptors. Furthermore, a *close()* followed by a reopen operation (e.g. *open()*, *dup()* etc) is not  
 2090    atomic; *dup2()* should be used to change standard file descriptors.

2091    *Rationale*: Austin Group Defect Report(s) applied: 555. See <http://austingroupbugs.net/view.php?id=555>.

2092    **Change Number: XSH/TC2/D4/0070** [810]

2093    On Page: 687 Line: 23229 Section: *confstr()*

2094    In the DESCRIPTION section, change from:

2095    the string stored in *buf* will contain the <space>-separated list of variable=value environment variable pairs  
 2096    required by the implementation to create a conforming environment, as described in the implementations'  
 2097    conformance documentation.

2098    to:

2099    the string stored in *buf* shall contain a <space>-separated list of the variable=value environment variable  
 2100    pairs an implementation requires as part of specifying a conforming environment, as described in the  
 2101    implementations' conformance documentation

2102    *Rationale*: Austin Group Defect Report(s) applied: 810. See <http://austingroupbugs.net/view.php?id=810>.

2103    **Change Number: XSH/TC2/D4/0071** [911]

2104    On Page: 687 Line: 23234 Section: *confstr()*  
 2105    (2013 edition Page: 692 Line: 23446)

2106    In the DESCRIPTION section, change from:

2107    can be used as a value of the *PATH* environment variable that accesses all of the standard utilities of  
 2108    POSIX.1-2008

2109    to:

2110    can be used as a value of the *PATH* environment variable that accesses all of the standard utilities of  
 2111    POSIX.1-2008 that are provided in a manner accessible via the *exec* family of functions

2112    *Rationale*: Austin Group Defect Report(s) applied: 911. See <http://austingroupbugs.net/view.php?id=911>.

2113    **Change Number: XSH/TC2/D4/0072** [630]

2114    On Page: 696 Line: 23586 Section: *cosh()*

2115    In the APPLICATION USAGE section, delete:

2116    For IEEE Std 754-1985 **double**,  $710.5 < |x|$  implies that *cosh(x)* has overflowed.

2117 *Rationale*: Austin Group Defect Report(s) applied: 630. See <http://austingroupbugs.net/view.php?id=630>.

2118 **Change Number: XSH/TC2/D4/0073** [899]

2119 On Page: 705 Line: 23827 Section: crypt()  
 2120 (2013 edition Page: 710 Line: 24043)

2121 In the APPLICATION USAGE section, add a new paragraph:

2122 Several implementations offer extensions via characters outside of the set specified for the *salt* argument  
 2123 for specifying alternative algorithms; while not portable, these extensions may offer better security. The use  
 2124 of *crypt()* for anything other than password hashing is not recommended.

2125 *Rationale*: Austin Group Defect Report(s) applied: 899. See <http://austingroupbugs.net/view.php?id=899>.

2126 **Change Number: XSH/TC2/D4/0074** [656]

2127 On Page: 713 Line: 24015 Section: ctermid()

2128 In the RETURN VALUE section, change from:

2129 If *s* is a null pointer, the string shall be generated in an area that may be static (and therefore may be  
 2130 overwritten by each call), the address of which shall be returned. Otherwise, ...

2131 to:

2132 If *s* is a null pointer, the string shall be generated in an area that may be static, the address of which shall be  
 2133 returned. The application shall not modify the string returned. The returned pointer might be invalidated or  
 2134 the string content might be overwritten by a subsequent call to *ctermid()*. The returned pointer might also  
 2135 be invalidated if the calling thread is terminated. If *s* is not a null pointer, ...

2136 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 2137 This item is a layered change on XSH/TC1/D5/0065 [75,428].

2138 The change is to add the following text to the end of the paragraph at 2013 edition P718, L24231-24234:

2139 The returned pointer might also be invalidated if the calling thread is terminated.

2140 **Change Number: XSH/TC2/D4/0075** [664]

2141 On Page: 715 Line: 24085 Section: ctime()

2142 In the DESCRIPTION section, change from:

2143 Unlike *ctime()*, the thread-safe version of *ctime\_r()* is not required to set *tzname*.

2144 to:

2145 If *ctime\_r()* sets *tzname*, it shall also set *daylight* and *timezone*. If *ctime\_r()* does not set *tzname*, it shall not  
 2146 set *daylight* and shall not set *timezone*.

2147 *Rationale*: Austin Group Defect Report(s) applied: 664. See <http://austingroupbugs.net/view.php?id=664>.  
2148 This change is layered upon XSH/TC1/D5/0066 [321,428]. The change replaces the last paragraph of the  
2149 description in 2013 edition, page 720 lines 24308-24309. The change is made for consistent behavior.

2150 **Change Number: XSH/TC2/D4/0076** [572]

2151 On Page: 723 Line: 24333-24354 Section: dirfd()

2152 In the DESCRIPTION section, at line 24333 change from:

2153 and may set *errno*

2154 to:

2155 and shall set *errno*

2156 In the ERRORS section, at line 24388 delete the ENOTSUP error.

2157 In the RATIONALE section, at line 24354 delete:

2158 An implementation that does not support file descriptors referring to directories may fail with [ENOTSUP].

2159 *Rationale*: Austin Group Defect Report(s) applied: 572. See <http://austingroupbugs.net/view.php?id=572>.  
2160 All implementations must support a file descriptor referring to a directory, even if that descriptor is not  
2161 used by readdir(). Furthermore, failure of dirfd() without setting errno is not useful to applications.

2162 **Change Number: XSH/TC2/D4/0077** [830]

2163 On Page: 725 Line: 24372 Section: dirname()  
2164 (2013 edition Page: 730 Line: 24593)

2165 In the DESCRIPTION section, after the text:

2166 ... return a pointer to a string that is a pathname of the parent directory of that file.

2167 add a sentence:

2168 The *dirname()* function shall not perform pathname resolution; the result shall not be affected by whether  
2169 or not *path* exists or by its file type.

2170 *Rationale*: Austin Group Defect Report(s) applied: 830. See <http://austingroupbugs.net/view.php?id=830>.  
2171 There is no requirement that the strings processed by basename() and dirname() refer to existing  
2172 pathnames.

2173 **Change Number: XSH/TC2/D4/0078** [612]

2174 On Page: 725 Line: 24372,24375 Section: dirname()

2175 In the DESCRIPTION section, at line 24372 change from:

2176 Trailing '/' characters in the path are not counted as part of the path.

2177 to:

2178 Trailing '/' characters in the path that are not also leading '/' characters shall not be counted as part of the  
 2179 path.

2180 Add a new paragraph after line 24375 (after paragraph two):

2181 The *dirname()* function may modify the string pointed to by *path*, and may return a pointer to static storage  
 2182 that may then be overwritten by a subsequent call to *dirname()*.

2183 *Rationale*: Austin Group Defect Report(s) applied: 612. See <http://austingroupbugs.net/view.php?id=612>.

2184 **Change Number: XSH/TC2/D4/0079** [830]

2185 On Page: 725 Line: 24378 Section: *dirname()*  
 2186 (2013 edition Page: 730 Line: 24599)

2187 In the RETURN VALUE section, change from:

2188 The *dirname()* function shall return a pointer to a string that is the parent directory of *path*. If *path* is a null  
 2189 pointer or points to an empty string, a pointer to a string "." is returned.

2190 to:

2191 The *dirname()* function shall return a pointer to a string as described above.

2192 *Rationale*: Austin Group Defect Report(s) applied: 830. See <http://austingroupbugs.net/view.php?id=830>.

2193 **Change Number: XSH/TC2/D4/0080** [656]

2194 On Page: 725 Line: 24380 Section: *dirname()*

2195 In the RETURN VALUE section, change from:

2196 The *dirname()* function may modify the string pointed to by *path*, and may return a pointer to static storage  
 2197 that may then be overwritten by subsequent calls to *dirname()*.

2198 to:

2199 The *dirname()* function may modify the string pointed to by *path*, and may return a pointer to internal  
 2200 storage. The returned pointer might be invalidated or the storage might be overwritten by a subsequent call  
 2201 to *dirname()*. The returned pointer might also be invalidated if the calling thread is terminated.

2202 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 2203 This item is a layered change on XSH/TC1/D5/0068 [75]

2204 The change is to add the following text to the end of the paragraph at 2013 edition P730, L24601-24603:

2205 The returned pointer might also be invalidated if the calling thread is terminated.

2206

2207 **Change Number: XSH/TC2/D4/0081** [612]

2208 On Page: 726 Line: 24404-24413 Section: dirname()

2209 In the EXAMPLES section, replace lines 24404-24413 with:

2210 The EXAMPLES section of the *basename()* function (see XREF basename() EXAMPLES section on page XXX) includes a table showing examples of the results of processing several sample pathnames by the *basename()* and *dirname()* functions and by the *basename* and *dirname* utilities.

2213 *Rationale:* Austin Group Defect Report(s) applied: 612. See <http://austingroupbugs.net/view.php?id=612>.

2214 **Change Number: XSH/TC2/D4/0082** [656]

2215 On Page: 730 Line: 24542 Section: dlerror()

2216 In the RETURN VALUE section, add a new paragraph at the end of the section:

2217 The application shall not modify the string returned. The returned pointer might be invalidated or the string content might be overwritten by a subsequent call to *dlerror()* in the same thread (if *dlerror()* is thread-safe) or in any thread (if *dlerror()* is not thread-safe). The returned pointer might also be invalidated if the calling thread is terminated.

2221 *Rationale:* Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
2222 This item is a layered change on XSH/TC1/D5/0070 [75] impacting the first part of that change.

2223 The change is to add the following text to the end of the paragraph at 2013 edition P735, L24774-24776:

2224 The returned pointer might also be invalidated if the calling thread is terminated.

2225 Note that the deletion of text on L24555 in XSH/TC1/D5/0070 is not impacted by this change.

2226 **Change Number: XSH/TC2/D4/0083** [743]

2227 On Page: 739 Line: 24845 Section: drand48()  
2228 (2013 edition Page: 744 Line: 25096)

2229 In the APPLICATION USAGE section, change from:

2230 None.

2231 to:

2232 These functions should be avoided whenever non-trivial requirements (including safety) have to be fulfilled.

2234 On Page: 739 Line: 24851 Section: drand48()  
2235 (2013 edition Page: 744 Line: 25102)

2236 In the SEE ALSO section, add *random()*.

2237 *Rationale:* Austin Group Defect Report(s) applied: 743. See <http://austingroupbugs.net/view.php?id=743>.

2238 **Change Number:** XSH/TC2/D4/0084 [753]

2239 On Page: 743 Line: 24972 Section: duplocale()

2240 In the EXAMPLES section, change from:

2241 `return retval; }`

2242 to:

2243 `return retval;`

2244 *Rationale:* Austin Group Defect Report(s) applied: 753. See <http://austingroupbugs.net/view.php?id=753>.

2245 **Change Number:** XSH/TC2/D4/0085 [899]

2246 On Page: 745 Line: 25023 Section: encrypt()  
 2247 (2013 edition Page: 750 Line: 25291)

2248 In the FUTURE DIRECTIONS section, change from:

2249 None.

2250 to:

2251 A future version of the standard may mark this interface as obsolete or remove it altogether.

2252 *Rationale:* Austin Group Defect Report(s) applied: 899. See <http://austingroupbugs.net/view.php?id=899>.

2253 **Change Number:** XSH/TC2/D4/0086 [493]

2254 On Page: 747 Line: 25044-25057 Section: endgrent()

2255 In the DESCRIPTION section, at line 25044 change from:

2256 When first called, `getgrent()` shall return a pointer to a **group** structure containing the first entry in the  
 2257 group database.

2258 to:

2259 If the group database is not already open, `getgrent()` shall open it and return a pointer to a **group** structure  
 2260 containing the first entry in the database.

2261 In the DESCRIPTION section, at line 25053 change from:

2262 The `setgrent()` function shall rewind the group database to allow repeated searches.

2263 to:

2264 The *setgrent()* function shall rewind the group database so that the next *getgrent()* call returns the first  
2265 entry, allowing repeated searches.

2266 In the DESCRIPTION section, at line 25054 change from:

2267 The *endgrent()* function may be called to close the group database when processing is complete.

2268 to:

2269 The *endgrent()* function shall close the group database.

2270 The *setgrent()* and *endgrent()* functions shall not change the setting of *errno* if successful.

2271 On error, the *setgrent()* and *endgrent()* functions shall set *errno* to indicate the error.

2272 Since no value is returned by the *setgrent()* and *endgrent()* functions, an application wishing to check for  
2273 error situations should set *errno* to 0, then call the function, then check *errno*.

2274 In the RETURN VALUE section, at line 25057 change from:

2275 When first called, *getgrent()* shall return a pointer to the first **group** structure in the group database. Upon  
2276 subsequent calls it shall return the next **group** structure in the group database. The *getgrent()* function shall  
2277 return a null pointer on end-of-file or an error and *errno* may be set to indicate the error.

2278 to:

2279 On successful completion, *getgrent()* shall return a pointer to a **group** structure. On end-of-file, *getgrent()*  
2280 shall return a null pointer and shall not change the setting of *errno*. On error, *getgrent()* shall return a null  
2281 pointer and *errno* shall be set to indicate the error.

2282 *Rationale:* Austin Group Defect Report(s) applied: 493. See <http://austingroupbugs.net/view.php?id=493>.  
2283 The changes are made for consistency.

2284 **Change Number: XSH/TC2/D4/0087** [656]

2285 On Page: 747 Line: 25061 Section: *endgrent()*

2286 In the RETURN VALUE section, change from:

2287 The return value may point to a static area which is overwritten by a subsequent call to *getgrgid()*,  
2288 *getgrnam()*, or *getgrent()*.

2289 to:

2290 The application shall not modify the structure to which the return value points, nor any storage areas  
2291 pointed to by pointers within the structure. The returned pointer, and pointers within the structure, might be  
2292 invalidated or the structure or the storage areas might be overwritten by a subsequent call to *getgrgid()*,  
2293 *getgrnam()*, or *getgrent()*. The returned pointer, and pointers within the structure, might also be invalidated  
2294 if the calling thread is terminated.

2295 *Rationale:* Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.

2296 This item is a layered change on XSH/TC1/D5/0080 [75].

2297 The change is to add the following text to the end of the paragraph at 2013 edition P752 L25329-25332:

2298 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
2299 terminated.

2300 **Change Number: XSH/TC2/D4/0088** [493]

2301 On Page: 747 Line: 25064 Section: endgrent()

2302 In the ERRORS section, change from:

2303 The *getgrent()* function may fail if:

2304 [EINTR] A signal was caught during the operation.

2305 [EIO] An I/O error has occurred.

2306 to:

2307 These functions may fail if:

2308 [EINTR] A signal was caught during the operation.

2309 [EIO] An I/O error has occurred.

2310 In addition, the *getgrent()* and *setgrent()* functions may fail if:

2311 *Rationale:* Austin Group Defect Report(s) applied: 493. See <http://austingroupbugs.net/view.php?id=493>.  
2312 The changes are made for consistency.

2313 **Change Number: XSH/TC2/D4/0089** [656]

2314 On Page: 749 Line: 25121 Section: endhostent()

2315 In the RETURN VALUE section, add a new paragraph to the end of the section:

2316 The application shall not modify the structure to which the return value points, nor any storage areas  
2317 pointed to by pointers within the structure. The returned pointer, and pointers within the structure, might be  
2318 invalidated or the structure or the storage areas might be overwritten by a subsequent call to *gethostent()*.  
2319 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
2320 terminated.

2321 *Rationale:* Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
2322 This item is a layered change on XSH/TC1/D5/0081 [75,428].

2323 The change is to add the following text to the end of the paragraph at 2013 edition P754, L25393-25396:

2324 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
2325 terminated.

2326 **Change Number: XSH/TC2/D4/0090** [656]

2327 On Page: 751 Line: 25175 Section: endnetent()

2328 In the RETURN VALUE section, add a new paragraph to the end of the section:

2329 The application shall not modify the structure to which the return value points, nor any storage areas  
2330 pointed to by pointers within the structure. The returned pointer, and pointers within the structure, might be  
2331 invalidated or the structure or the storage areas might be overwritten by a subsequent call to *getnetbyaddr()*,  
2332 *getnetbyname()*, or *getnetent()*. The returned pointer, and pointers within the structure, might also be  
2333 invalidated if the calling thread is terminated.

2334 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
2335 This item is a layered change on XSH/TC1/D5/0083 [75].

2336 The change is to add the following text to the end of the paragraph at 2013 edition P756, L25452-25455:

2337 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
2338 terminated.

2339 **Change Number: XSH/TC2/D4/0091** [656]

2340 On Page: 753 Line: 25230 Section: endprotoent()

2341 In the RETURN VALUE section, add a new paragraph to the end of the section:

2342 The application shall not modify the structure to which the return value points, nor any storage areas  
2343 pointed to by pointers within the structure. The returned pointer, and pointers within the structure, might be  
2344 invalidated or the structure or the storage areas might be overwritten by a subsequent call to  
2345 *getprotobynumber()*, *getprotobyname()*, or *getprotoent()*. The returned pointer, and pointers within the  
2346 structure, might also be invalidated if the calling thread is terminated.

2347 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
2348 This item is a layered change on XSH/TC1/D5/0085 [75].

2349 The change is to add the following text to the end of the paragraph at 2013 edition P758, L25512-25515:

2350 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
2351 terminated.

2352 **Change Number: XSH/TC2/D4/0092** [493]

2353 On Page: 755 Line: 25259-25271 Section: endpwent()

2354 In the DESCRIPTION section, at line 25259 change from:

2355 When first called, *getpwent()* shall return a pointer to a **passwd** structure containing the first entry in the  
2356 user database.

2357 to:

2358 If the user database is not already open, *getpwent()* shall open it and return a pointer to a **passwd** structure  
 2359 containing the first entry in the database.

2360 In the DESCRIPTION section, at line 25267 change from:

2361 The *setpwent()* function effectively rewinds the user database to allow repeated searches.

2362 to:

2363 The *setpwent()* function shall rewind the user database so that the next *getpwent()* call returns the first  
 2364 entry, allowing repeated searches.

2365 In the DESCRIPTION section, at line 25268 change from:

2366 The *endpwent()* function may be called to close the user database when processing is complete.

2367 to:

2368 The *endpwent()* function shall close the user database.

2369 The *setpwent()* and *endpwent()* functions shall not change the setting of *errno* if successful.

2370 On error, the *setpwent()* and *endpwent()* functions shall set *errno* to indicate the error.

2371 Since no value is returned by the *setpwent()* and *endpwent()* functions, an application wishing to check for  
 2372 error situations should set *errno* to 0, then call the function, then check *errno*.

2373 In the RETURN VALUE section, at line 25271 change from:

2374 The *getpwent()* function shall return a null pointer on end-of-file or error.

2375 to:

2376 On successful completion, *getpwent()* shall return a pointer to a **passwd** structure. On end-of-file,  
 2377 *getpwent()* shall return a null pointer and shall not change the setting of *errno*. On error, *getpwent()* shall  
 2378 return a null pointer and *errno* shall be set to indicate the error.

2379 *Rationale:* Austin Group Defect Report(s) applied: 493. See <http://austingroupbugs.net/view.php?id=493>.  
 2380 The changes are made for consistency.

2381 **Change Number: XSH/TC2/D4/0093 [656]**

2382 On Page: 755 Line: 25271 Section: *endpwent()*

2383 In the RETURN VALUE section, add a new paragraph to the end of the section:

2384 The application shall not modify the structure to which the return value points, nor any storage areas  
 2385 pointed to by pointers within the structure. The returned pointer, and pointers within the structure, might be  
 2386 invalidated or the structure or the storage areas might be overwritten by a subsequent call to *getpwuid()*,  
 2387 *getpwnam()*, or *getpwent()*. The returned pointer, and pointers within the structure, might also be  
 2388 invalidated if the calling thread is terminated.

2389 *Rationale:* Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
2390 This item is a layered change on XSH/TC1/D5/0087 [75].

2391 The change is to add the following text to the end of the paragraph at 2013 edition P760, L25558-25561:

2392 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
2393 terminated.

2394 Note that the deletion of text on L25278 in XSH/TC1/D5/0087 is not impacted by this change.

2395 **Change Number:** XSH/TC2/D4/0094 [493]

2396 On Page: 755 Line: 25274 Section: endpwent()

2397 In the ERRORS section, insert before the EIO error:

2398 [EINTR] A signal was caught during the operation.

2399 *Rationale:* Austin Group Defect Report(s) applied: 493. See <http://austingroupbugs.net/view.php?id=493>.  
2400 The changes are made for consistency.

2401 **Change Number:** XSH/TC2/D4/0095 [656]

2402 On Page: 758 Line: 25362 Section: endservent()

2403 In the RETURN VALUE section, add a new paragraph to the end of the section:

2404 The application shall not modify the structure to which the return value points, nor any storage areas  
2405 pointed to by pointers within the structure. The returned pointer, and pointers within the structure, might be  
2406 invalidated or the structure or the storage areas might be overwritten by a subsequent call to  
2407 *getservbyname()*, *getservbyport()*, or *getservent()*. The returned pointer, and pointers within the structure,  
2408 might also be invalidated if the calling thread is terminated.

2409 *Rationale:* Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
2410 This item is a layered change on XSH/TC1/D5/0088 [75].

2411 The change is to add the following text to the end of the paragraph at 2013 edition, P763, L25652-25655:

2412 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
2413 terminated.

2414 **Change Number:** XSH/TC2/D4/0096 [630]

2415 On Page: 767 Line: 25598 Section: erfc()

2416 In the APPLICATION USAGE section, delete:

2417 Note for IEEE Std 754-1985 **double**,  $26.55 < x$  implies *erfc(x)* has underflowed.

2418 *Rationale:* Austin Group Defect Report(s) applied: 630. See <http://austingroupbugs.net/view.php?id=630>.

2419 **Change Number: XSH/TC2/D4/0097** [584]

2420 On Page: 780 Line: 26023 Section: exec

2421 In the RATIONALE section, change from:

2422 <hyphen>

2423 to:

2424 <hyphen-minus>

2425 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

2426 **Change Number: XSH/TC2/D4/0098** [898]

2427 On Page: 780 Line: 26024 Section: exec

2428 (2013 edition Page: 785 Line: 26351)

2429 In the RATIONALE section, add a new paragraph:

2430 Also, note that the *test* and *[* utilities require specific strings for the *argv[0]* argument to have deterministic  
 2431 behavior across all implementations.

2432 *Rationale*: Austin Group Defect Report(s) applied: 898. See <http://austingroupbugs.net/view.php?id=898>.

2433 **Change Number: XSH/TC2/D4/0099** [734]

2434 On Page: 782 Line: 26142 Section: exec

2435 (2013 edition Page: 788 Line: 26469)

2436 In the RATIONALE section, change from:

2437 Alternatively, a function like *openat()*

2438 to:

2439 Alternatively, a function like *openat()*

2440 *Rationale*: Austin Group Defect Report(s) applied: 734. See <http://austingroupbugs.net/view.php?id=734>.

2441 **Change Number: XSH/TC2/D4/0100** [594]

2442 On Page: 785 Line: 26213 Section: *exit()*

2443 In the DESCRIPTION section, change from:

2444 the least significant 8 bits (that is, *status* & 0377) shall be available to a waiting parent process

2445 to:

2446 the least significant 8 bits (that is, *status* & 0377) shall be available from *wait()* and *waitpid()*; the full value  
 2447 shall be available from *waitid()* and in the **siginfo\_t** passed to a signal handler for SIGCHLD.

2448 *Rationale*: Austin Group Defect Report(s) applied: 594. See <http://austingroupbugs.net/view.php?id=594>.  
 2449 The standard is unclear about when *si\_status* contains an exit status and when it contains a signal.

2450 **Change Number: XSH/TC2/D4/0101** [630]

2451 On Page: 787 Line: 26297 Section: *exp()*

2452 In the APPLICATION USAGE section, delete:

2453 Note that for IEEE Std 754-1985 **double**,  $709.8 < x$  implies *exp(x)* has overflowed. The value  $x > 708.4$   
 2454 implies *exp(x)* has underflowed.

2455 *Rationale*: Austin Group Defect Report(s) applied: 630. See <http://austingroupbugs.net/view.php?id=630>.

2456 **Change Number: XSH/TC2/D4/0102** [630]

2457 On Page: 789 Line: 26366 Section: *exp2()*

2458 In the APPLICATION USAGE section, delete:

2459 For IEEE Std 754-1985 **double**,  $1024 \leq x$  implies *exp2(x)* has overflowed. The value  $x < -1022$  implies  
 2460 *exp(x)* has underflowed.

2461 *Rationale*: Austin Group Defect Report(s) applied: 630. See <http://austingroupbugs.net/view.php?id=630>.

2462 **Change Number: XSH/TC2/D4/0103** [630]

2463 On Page: 791 Line: 26426 Section: *expm1()*

2464 In the APPLICATION USAGE section, delete:

2465 For IEEE Std 754-1985 **double**,  $709.8 < x$  implies *expm1(x)* has overflowed.

2466 *Rationale*: Austin Group Defect Report(s) applied: 630. See <http://austingroupbugs.net/view.php?id=630>.

2467 **Change Number: XSH/TC2/D4/0104** [555]

2468 On Page: 806 Line: 26839 Section: *fclose()*

2469 In the APPLICATION USAGE section, change from:

2470 None.

2471 to:

2472 Since after the call to *fclose()* any use of *stream* results in undefined behavior, *fclose()* should not be used  
 2473 on *stdin*, *stdout*, or *stderr* except immediately before process termination (see XBD 3.297 on page 81), so  
 2474 as to avoid triggering undefined behavior in other standard interfaces that rely on these streams. If there are

2475 any *atexit()* handlers registered by the application, such a call to *fclose()* should not occur until the last  
 2476 handler is finishing. Once *fclose()* has been used to close *stdin*, *stdout*, or *stderr*, there is no standard way to  
 2477 reopen any of these streams.

2478 Use of *freopen()* to change *stdin*, *stdout*, or *stderr* instead of closing them avoids the danger of a file  
 2479 unexpectedly being opened as one of the special file descriptors *STDIN\_FILENO*, *STDOUT\_FILENO*, or  
 2480 *STDERR\_FILENO* at a later time in the application.

2481 *Rationale*: Austin Group Defect Report(s) applied: 555. See <http://austingroupbugs.net/view.php?id=555>.

2482 **Change Number: XSH/TC2/D4/0105** [835]

2483 On Page: 807 Line: 26877 Section: *fcntl()*  
 2484 (2013 edition Page: 813 Line: 27229)

2485 Change from:

2486 Return a new file descriptor which shall be the lowest numbered available (that is, not already open) file  
 2487 descriptor greater than or equal to the third argument, *arg*, taken as an integer of type **int**.

2488 to:

2489 Return a new file descriptor which shall be allocated as described in [xref to new section 2.14] except that  
 2490 it shall be the lowest numbered available file descriptor greater than or equal to the third argument, *arg*,  
 2491 taken as an integer of type **int**.

2492 *Rationale*: Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

2493 **Change Number: XSH/TC2/D4/0106** [677]

2494 On Page: 807 Line: 26911-26920 Section: *fcntl()*

2495 In the DESCRIPTION section, replace the *F\_GETOWN* and *F\_SETOWN* descriptions with:

2496 ***F\_GETOWN***

2497 If *fd* refers to a socket, get the process ID or process group ID specified to receive SIGURG  
 2498 signals when out-of-band data is available. Positive values shall indicate a process ID; negative  
 2499 values, other than -1, shall indicate a process group ID; the value zero shall indicate that no  
 2500 SIGURG signals are to be sent. If *fd* does not refer to a socket, the results are unspecified.

2501 ***F\_SETOWN***

2502 If *fd* refers to a socket, set the process ID or process group ID specified to receive SIGURG  
 2503 signals when out-of-band data is available, using the value of the third argument, *arg*, taken as type  
 2504 **int**. Positive values shall indicate a process ID; negative values, other than -1, shall indicate a  
 2505 process group ID; the value zero shall indicate that no SIGURG signals are to be sent. Each time a  
 2506 SIGURG signal is sent to the specified process or process group, permission checks equivalent to  
 2507 those performed by *kill()* shall be performed, as if *kill()* were called by a process with the same real  
 2508 user ID, effective user ID, and privileges that the process calling *fcntl()* has at the time of the call; if  
 2509 the *kill()* call would fail, no signal shall be sent. These permission checks may also be performed by  
 2510 the *fcntl()* call. If the process specified by *arg* later terminates, or the process group specified by *arg*  
 2511 later becomes empty, while still being specified to receive SIGURG signals when out-of-band data  
 2512 is available from *fd*, then no signals shall be sent to any subsequently created process that has the  
 2513 same process ID or process group ID, regardless of permission; it is unspecified whether this is

2514        achieved by the equivalent of an `fcntl(fd, F_SETOWN, 0)` call at the time the process  
2515        terminates or is waited for or the process group becomes empty, or by other means. If `fd` does not  
2516        refer to a socket, the results are unspecified.

2517        *Rationale:* Austin Group Defect Report(s) applied: 677. See <http://austingroupbugs.net/view.php?id=677>.

2518        **Change Number: XSH/TC2/D4/0107** [484]

2519        On Page: 808 Line: 26923 Section: `fcntl()`

2520        In the DESCRIPTION section, for `F_GETLK` change from:

2521        Get the first lock which blocks ...

2522        to:

2523        Get any lock which blocks ...

2524        *Rationale:* Austin Group Defect Report(s) applied: 484. See <http://austingroupbugs.net/view.php?id=484>.  
2525        It was unclear whether "first lock" means a lock with the lowest start offset or a first acquired lock in a  
2526        chronological order or even something completely different.

2527        **Change Number: XSH/TC2/D4/0108** [675]

2528        On Page: 810 Line: 27040 Section: `fcntl()`  
2529        (2013 edition Page: 816 Line: 27394)

2530        In the ERRORS section add after the [EOVERFLOW] error:

2531        **[ESRCH]**

2532        The `cmd` argument is `F_SETOWN` and no process or process group can be found corresponding to  
2533        that specified by `arg`.

2534        *Rationale:* Austin Group Defect Report(s) applied: 675. See <http://austingroupbugs.net/view.php?id=675>.

2535        **Change Number: XSH/TC2/D4/0109** [675,677]

2536        On Page: 811 Line: 27044 Section: `fcntl()`

2537        In the ERRORS section, add to the "may fail" errors:

2538        **[EINVAL]**

2539        The `cmd` argument is `F_SETOWN` and the value of the argument is not valid as a process or process  
2540        group identifier.

2541        **[EPERM]**

2542        The `cmd` argument is `F_SETOWN` and the calling process does not have permission to send a  
2543        `SIGURG` signal to any process specified by `arg`.

2544        On Page: 812 Line: 27107 Section: `fcntl()`

2545 In the APPLICATION USAGE section, add a new paragraph:

2546 On systems which do not perform permission checks at the time of an *fcntl()* call with F\_SETOWN, if the  
 2547 permission checks performed at the time the signal is sent disallow sending the signal to any process, the  
 2548 process that called *fcntl()* has no way of discovering that this has happened. A call to *kill()* with signal 0 can  
 2549 be used as a prior check of permissions, although this is no guarantee that permission will be granted at the  
 2550 time a signal is sent, since the target process(es) could change user IDs or privileges in the meantime.

2551 On Page: 814 Line: 27172 Section: *fcntl()*

2552 In the SEE ALSO section, add *kill()*.

2553 *Rationale*: Austin Group Defect Report(s) applied: 675,677. See  
<http://austingroupbugs.net/view.php?id=675>.  
 2554 See <http://austingroupbugs.net/view.php?id=677>.

2555 **Change Number: XSH/TC2/D4/0110** [501]

2556 On Page: 815 Line: 27225 Section: *fdasynchr()*

2557 In the ERRORS section, change from:

2558 The *fd* argument is not a valid file descriptor open for writing.

2559 to:

2560 The *fd* argument is not a valid file descriptor.

2561 At line 27232 in the APPLICATION USAGE section, change from:

2562 None.

2563 to:

2564 Note that even if the file descriptor is not open for writing, if there are any pending write requests on the  
 2565 underlying file, then that I/O will be completed prior to the return of *fdasynchr()*.

2566 *Rationale*: Austin Group Defect Report(s) applied: 501. See <http://austingroupbugs.net/view.php?id=501>.  
 2567 The access mode of the file descriptor does not affect whether there are pending I/O operations on the  
 2568 underlying file.

2569 **Change Number: XSH/TC2/D4/0111** [543]

2570 On Page: 834 Line: 27835 Section: *feraiseexcept()*

2571 In the DESCRIPTION section, change from:

2572 The order in which these floating-point exceptions are raised is unspecified.

2573 to:

2575 The order in which these floating-point exceptions are raised is unspecified, [MX]except that if the *excepts*  
2576 argument represents IEC 60559 valid coincident floating-point exceptions for atomic operations (namely  
2577 overflow and inexact, or underflow and inexact), then overflow or underflow shall be raised before  
2578 inexact.[/MX]

2579 *Rationale*: Austin Group Defect Report(s) applied: 543. See <http://austingroupbugs.net/view.php?id=543>.

2580 **Change Number: XSH/TC2/D4/0112** [816]

2581 On Page: 844 Line: 28021 Section: *fflush()*

2582 In the DESCRIPTION section, moves lines 28021-28022 to become the final paragraph:

2583 If *stream* is a null pointer, *fflush()* shall perform this flushing action on all streams for which the behavior is  
2584 defined above.

2585 *Rationale*: Austin Group Defect Report(s) applied: 816. See <http://austingroupbugs.net/view.php?id=816>.  
2586 This makes it clear that *fflush(NULL)* must affect underlying positions of seekable fds associated with read  
2587 streams.

2588 **Change Number: XSH/TC2/D4/0113** [626]

2589 On Page: 844 Line: 28023 Section: *fflush()*

2590 In the DESCRIPTION section, change from:

2591 For a stream open for reading,

2592 to:

2593 For a stream open for reading with an underlying file description,

2594 *Rationale*: Austin Group Defect Report(s) applied: 626. See <http://austingroupbugs.net/view.php?id=626>.

2595 **Change Number: XSH/TC2/D4/0114** [468]

2596 On Page: 852 Line: 28283-28285 Section: *fgets()*

2597 In the DESCRIPTION section, change from:

2598 The *fgets()* function shall read bytes from *stream* into the array pointed to by *s*, until *n*-1 bytes are read, or a  
2599 <newline> is read and transferred to *s*, or an end-of-file condition is encountered. The string is then  
2600 terminated with a null byte.

2601 to:

2602 The *fgets()* function shall read bytes from *stream* into the array pointed to by *s* until *n*-1 bytes are read, or a  
2603 <newline> is read and transferred to *s*, or an end-of-file condition is encountered. A null byte shall be  
2604 written immediately after the last byte read into the array. If the end-of-file condition is encountered before  
2605 any bytes are read, the contents of the array pointed to by *s* shall not be changed.

2606 *Rationale*: Austin Group Defect Report(s) applied: 468. See <http://austingroupbugs.net/view.php?id=468>.  
 2607 This is a conflict with the C99 standard. All current implementations are thought to implement this as C99  
 2608 requires.

2609 **Change Number: XSH/TC2/D4/0115** [589]

2610 On Page: 858 Line: 28462-28464 Section: fileno()

2611 In the ERRORS section, replace the entire section with:

2612 The *fileno()* function shall fail if:

2613 [EBADF]

2614 The stream is not associated with a file.

2615 The *fileno()* function may fail if:

2616 [EBADF]

2617 The file descriptor underlying *stream* is not a valid file descriptor.

2618 *Rationale*: Austin Group Defect Report(s) applied: 589. See <http://austingroupbugs.net/view.php?id=589>.

2619 **Change Number: XSH/TC2/D4/0116** [611]

2620 On Page: 859 Line: 28523 Section: flockfile()

2621 In the APPLICATION USAGE section, add a new paragraph to the end of the section:

2622 A call to *exit()* can block until locked streams are unlocked because a thread having ownership of a (**FILE**  
 2623 \*) object blocks all function calls that reference that (**FILE** \*) object (except those with names ending in  
 2624 \_unlocked) from other threads, including calls to *exit()*.

2625 *Rationale*: Austin Group Defect Report(s) applied: 611. See <http://austingroupbugs.net/view.php?id=611>.

2626 **Change Number: XSH/TC2/D4/0117** [587]

2627 On Page: 866 Line: 28763 Section: fmemopen()

2628 In the DESCRIPTION section, change from:

2629 for modes *a* and *a+* the initial size shall be either the position of the first null byte in the buffer or the value  
 2630 of the *size* argument if no null byte is found

2631 to:

2632 for modes *a* and *a+* the initial size shall be:

- 2633     • zero, if *buf* is a null pointer
- 2634     • the position of the first null byte in the buffer, if one is found
- 2635     • the value of the *size* argument, if *buf* is not a null pointer and no null byte is found

2636 *Rationale*: Austin Group Defect Report(s) applied: 587. See <http://austingroupbugs.net/view.php?id=587>.  
2637 The standard does not speak to the null pointer case, but existing practice handles this case consistently.

2638 **Change Number: XSH/TC2/D4/0118** [586,818]

2639 On Page: 867 Line: 28786,28790 Section: fmemopen()

2640 In the ERRORS section, on line 28786 change from:

2641 [EINVAL]

2642 The *size* argument specifies a buffer size of zero.

2643 to:

2644 [EMFILE]

2645 {STREAM\_MAX} streams are currently open in the calling process.

2646 On line 28790, add a new paragraph before the "may fail" ENOMEM error:

2647 [EINVAL]

2648 The *size* argument specifies a buffer size of zero and the implementation does not support this.

2649 *Rationale*: Austin Group Defect Report(s) applied: 586,818. See

2650 <http://austingroupbugs.net/view.php?id=586>.

2651 See <http://austingroupbugs.net/view.php?id=818>.

2652 **Change Number: XSH/TC2/D4/0119** [818]

2653 On Page: 868 Line: 28284 Section: fmemopen()

2654 In the FUTURE DIRECTIONS section, change from:

2655 None.

2656 to:

2657 A future revision of this standard may require support of zero length buffer streams explicitly.

2658 *Rationale*: Austin Group Defect Report(s) applied: 818. See <http://austingroupbugs.net/view.php?id=818>.

2659 **Change Number: XSH/TC2/D4/0120** [605]

2660 On Page: 870 Line: 28887 Section: fmod()

2661 In the RETURN VALUE section, change from:

2662 If *x* or *y* is NaN, a NaN shall be returned.

2663 to:

2664 If *x* or *y* is NaN, a NaN shall be returned, and none of the conditions below shall be considered.

2665 *Rationale*: Austin Group Defect Report(s) applied: 605. See <http://austingroupbugs.net/view.php?id=605>.

2666 **Change Number: XSH/TC2/D4/0121** [806]

2667 On Page: 875 Line: 29060 Section: fnmatch()

2668 In the DESCRIPTION section, change from:

2669 In particular, "\\" shall match a <backslash> in *string*.

2670 to:

2671 In particular, "\\" shall match a <backslash> in *string*. If *pattern* ends with an unescaped <backslash>,  
 2672 *fnmatch*() shall return a non-zero value (indicating either no match or an error).

2673 *Rationale*: Austin Group Defect Report(s) applied: 806. See <http://austingroupbugs.net/view.php?id=806>.  
 2674 Existing practice is for a pattern ending in an unescaped <backslash> not to match anything, but it would  
 2675 be useful to allow implementations to diagnose this as user error.

2676 **Change Number: XSH/TC2/D4/0122** [822]

2677 On Page: 878 Line: 29180 Section: fopen()

2678 In the ERRORS section, change from:

2679 [ENOENT]

2680 A component of *filename* does not name an existing file or *filename* is an empty string.

2681 to:

2682 [ENOENT]

2683 The *mode* string begins with 'x' and a component of *pathname* does not name an existing file, or  
 2684 mode begins with 'w' or 'a' and a component of the path prefix of *pathname* does not name an  
 2685 existing file, or *pathname* is an empty string.

2686 [ENOENT] or [ENOTDIR]

2687 The *pathname* argument contains at least one non-<slash> character and ends with one or more  
 2688 trailing <slash> characters. If *pathname* without the trailing <slash> characters would name an  
 2689 existing file, an [ENOENT] error shall not occur.

2690 *Rationale*: Austin Group Defect Report(s) applied: 822. See <http://austingroupbugs.net/view.php?id=822>.  
 2691 This change layers upon XSH/TC1/D5/0157 [146,433] changing the last sentence of the change.

2692 **Change Number: XSH/TC2/D4/0123** [858]

2693 On Page: 882 Line: 29311 Section: fork()  
 2694 (2013 edition Page: 889 Line: 29741)

2695 In the DESCRIPTION section, delete:

2696 Fork handlers may be established by means of the *pthread\_atfork*() function in order to maintain

2697 application invariants across *fork()* calls.

2698 *Rationale*: Austin Group Defect Report(s) applied: 858. See <http://austingroupbugs.net/view.php?id=858>.

2699 **Change Number: XSH/TC2/D4/0124** [651]

2700 On Page: 888 Line: 29550-29551 Section: fpathconf()

2701 In the ERRORS section, move the "shall fail" ELOOP error to the "may fail" part of the ERRORS section  
2702 before the other ELOOP error at line 29558.

2703 *Rationale*: Austin Group Defect Report(s) applied: 651. See <http://austingroupbugs.net/view.php?id=651>.  
2704 This was inconsistent with the requirements for other pathconf() error conditions.

2705 **Change Number: XSH/TC2/D4/0125** [651]

2706 On Page: 889 Line: 29612 Section: fpathconf()

2707 In the RATIONALE section, change from:

2708 ... not required to detect any of the errors except the meaning of [EINVAL] that indicates that the value of  
2709 *name* is not valid for that variable.

2710 to:

2711 ... not required to detect any of the errors except the meaning of [EINVAL] that indicates that the value of  
2712 *name* is not valid for that variable, and the [EOVERFLOW] error that indicates the value to be returned is  
2713 larger than {LONG\_MAX}.

2714 *Rationale*: Austin Group Defect Report(s) applied: 651. See <http://austingroupbugs.net/view.php?id=651>.

2715 **Change Number: XSH/TC2/D4/0126** [894]

2716 On Page: 895 Line: 29808 Section: fprintf()  
2717 (2013 edition Page: 902 Line: 30242)

2718 In the DESCRIPTION section, change from:

2719 For *o* conversion, it increases the precision (if necessary) to force the first digit of the result to be zero.

2720 to:

2721 For *o* conversion, it shall increase the precision, if and only if necessary, to force the first digit of the result  
2722 to be a zero (if the value and precision are both 0, a single 0 is printed).

2723 *Rationale*: Austin Group Defect Report(s) applied: 894. See <http://austingroupbugs.net/view.php?id=894>.

2724

2725 **Change Number: XSH/TC2/D4/0127** [557]

2726 On Page: 899 Line: 30005 Section: fprintf()

2727 In the ERRORS section, delete:

2728 In addition, all forms of *fprintf()* may fail if:

2729 [EINVAL] There are insufficient arguments.

2730 In the RATIONALE section, on page 904 line 30225 change from:

2731 None

2732 to:

2733 If an implementation detects that there are insufficient arguments for the format, it is recommended that the  
 2734 function should fail and report an [EINVAL] error.

2735 *Rationale:* Austin Group Defect Report(s) applied: 557. See <http://austingroupbugs.net/view.php?id=557>.  
 2736 The descriptions of *fprintf()* and *fwprintf()* say "The results are undefined if there are insufficient  
 2737 arguments for the format." However, the ERRORS section on each of those pages also specifies a "may  
 2738 fail" EINVAL error for this condition.

2739 **Change Number: XSH/TC2/D4/0128** [936]

2740 On Page: 904 Line: 30230 Section: fprintf()  
 2741 (2013 edition Page: 912 Line: 30677)

2742 In the SEE ALSO section, add <inttypes.h> to the list.

2743 *Rationale:* Austin Group Defect Report(s) applied: 936. See <http://austingroupbugs.net/view.php?id=936>.

2744 **Change Number: XSH/TC2/D4/0129** [926]

2745 On Page: 913 Line: 30537 Section: fread()  
 2746 (2013 edition Page: 921 Line: 31022)

2747 In the EXAMPLES section, change from:

2748 The following example reads a single element from the *fp* stream into the array pointed to by *buf*.

2749 to:

2750 The following example transfers a single 100-byte fixed length record from the *fp* stream into the array  
 2751 pointed to by *buf*.

2752 *Rationale:* Austin Group Defect Report(s) applied: 926. See <http://austingroupbugs.net/view.php?id=926>.

2753

2754 **Change Number: XSH/TC2/D4/0130** [939]

2755 On Page: 916 Line: 30649 Section: freeaddrinfo()  
 2756 (2013 edition Page: 924 Line: 31149)

2757 In the DESCRIPTION section, change from:

2758 In this *hints* structure every member other than *ai\_flags*, *ai\_family*, *ai\_socktype*, and *ai\_protocol* shall be  
 2759 set to zero or a null pointer.

2760 to:

2761 The application shall ensure that each of the *ai\_addr*, *ai\_canonname*, and *ai\_next* members, as  
 2762 well as each of the non-standard additional members, if any, of this *hints* structure is initialized. If any of  
 2763 these members has a value other than the value that would result from default initialization, the behavior is  
 2764 implementation-defined.

2765 *Rationale*: Austin Group Defect Report(s) applied: 939. See <http://austingroupbugs.net/view.php?id=939>.

2766 **Change Number: XSH/TC2/D4/0131** [979]

2767 On Page: 917 Line: 30683 Section: freeaddrinfo()  
 2768 (2013 edition Page: 925 Line: 31183)

2769 In the DESCRIPTION section, change from:

2770 If the AI\_V4MAPPED flag is specified along with an *ai\_family* of AF\_INET6 then *getaddrinfo()* shall  
 2771 return IPv4-mapped IPv6 addresses on finding no matching IPv6 addresses (*ai\_addr* shall be 16). The  
 2772 AI\_V4MAPPED flag shall be ignored unless *ai\_family* equals AF\_INET6. If the AI\_ALL flag is used with  
 2773 the AI\_V4MAPPED flag, then *getaddrinfo()* shall return all matching IPv6 and IPv4 addresses. The  
 2774 AI\_ALL flag without the AI\_V4MAPPED flag is ignored.

2775 to (keeping the IP6 shading):

2776 By default, with an *ai\_family* of AF\_INET6, *getaddrinfo()* shall return only IPv6 addresses. If the  
 2777 AI\_V4MAPPED flag is specified along with an *ai\_family* of AF\_INET6, then *getaddrinfo()* shall return  
 2778 IPv4-mapped IPv6 addresses on finding no matching IPv6 addresses. The AI\_V4MAPPED flag shall be  
 2779 ignored unless *ai\_family* equals AF\_INET6. If the AI\_ALL flag is used with the AI\_V4MAPPED flag,  
 2780 then *getaddrinfo()* shall return all matching IPv6 and IPv4 addresses. The AI\_ALL flag without the  
 2781 AI\_V4MAPPED flag shall be ignored.

2782 *Rationale*: Austin Group Defect Report(s) applied: 979. See <http://austingroupbugs.net/view.php?id=979>.

2783 **Change Number: XSH/TC2/D4/0132** [918]

2784 On Page: 919 Line: 30762 Section: freeaddrinfo()  
 2785 (2013 edition Page: 927 Line: 31262)

2786 In the EXAMPLES section, change from:

2787 `struct addrinfo hints = {};`

2788 to:

2789 `struct addrinfo hints = {0};`

2790 *Rationale:* Austin Group Defect Report(s) applied: 918. See <http://austingroupbugs.net/view.php?id=918>.

2791 **Change Number: XSH/TC2/D4/0133** [934]

2792 On Page: 920 Line: 30796 Section: freeaddrinfo()  
 2793 (2013 edition Page: 928 Line: 31296)

2794 In the APPLICATION USAGE section, add a new paragraph:

2795 Although it is common practice to initialize the *hints* structure using:

2796 `struct addrinfo hints;`  
 2797 `memset(&hints, 0, sizeof hints);`

2798 this method is not portable according to this standard, because the structure can contain pointer or floating  
 2799 point members that are not required to have an all-bits-zero representation after default initialization.  
 2800 Portable methods make use of default initialization, for example:

2801 `struct addrinfo hints = { 0 };`

2802 or

2803 `static struct addrinfo hints_init;`  
 2804 `struct addrinfo hints = hints_init;`

2805 A future version of this standard may require that a pointer object with an all-bits-zero representation is a  
 2806 null pointer, and that **addrinfo** does not have any floating-point members if a floating-point object with an  
 2807 all-bits-zero representation does not have the value 0.0.

2808 *Rationale:* Austin Group Defect Report(s) applied: 934. See <http://austingroupbugs.net/view.php?id=934>.

2809 **Change Number: XSH/TC2/D4/0134** [822]

2810 On Page: 924 Line: 30923 Section: freopen()

2811 In the ERRORS section, change from:

2812 `[ENOENT]`

2813 A component of filename does not name an existing file or filename is an empty string.

2814 to:

2815 `[ENOENT]`

2816 The *mode* string begins with 'r' and a component of pathname does not name an existing file, or  
 2817 mode begins with 'w' or 'a' and a component of the path prefix of *pathname* does not name an  
 2818 existing file, or *pathname* is an empty string.

2819 `[ENOENT]` or `[ENOTDIR]`

2820        The pathname argument contains at least one non-<slash> character and ends with one or more  
2821        trailing <slash> characters. If pathname without the trailing <slash> characters would name an  
2822        existing file, an [ENOENT] error shall not occur.

2823        *Rationale:* Austin Group Defect Report(s) applied: 822. See <http://austingroupbugs.net/view.php?id=822>.  
2824        This change layers upon XSH/TC1/D5/0182 [146,433], changing the last sentence of the change.

2825        **Change Number: XSH/TC2/D4/0135** [936]

2826        On Page: 935 Line: 31352 Section: fscanf()  
2827        (2013 edition Page: 944 Line: 31867)

2828        In the SEE ALSO section, add <inttypes.h> to the list.

2829        *Rationale:* Austin Group Defect Report(s) applied: 936. See <http://austingroupbugs.net/view.php?id=936>.

2830        **Change Number: XSH/TC2/D4/0136** [591]

2831        On Page: 945 Line: 31644 Section: fstatat()

2832        Before the fstatat SYNOPSIS line, insert a line with OH shading:

2833        `#include <fcntl.h>`

2834        *Rationale:* Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

2835        **Change Number: XSH/TC2/D4/0137** [817]

2836        On Page: 945 Line: 31686 Section: fstatat()  
2837        (2013 edition Page: 954 Line: 32215)

2838        In the DESCRIPTION section, change from:

2839        If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
2840        are permitted using the current permissions of the directory underlying the file descriptor. If the file  
2841        descriptor was opened with O\_SEARCH, the function shall not perform the check.

2842        to:

2843        If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
2844        function shall check whether directory searches are permitted using the current permissions of the directory  
2845        underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

2846        *Rationale:* Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

2847        **Change Number: XSH/TC2/D4/0138** [817]

2848        On Page: 946 Line: 31717 Section: fstatat()  
2849        (2013 edition Page: 955 Line: 32247)

2850        In the ERRORS section, for the [EACCES] error, change from:

2851    *fd* was not opened with O\_SEARCH and ...

2852    to:

2853    The access mode of the open file description associated with *fd* is not O\_SEARCH and ...

2854    *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

2855    **Change Number: XSH/TC2/D4/0139** [889]

2856    On Page: 948 Line: 31811 Section: fstatat()  
 2857    (2013 edition Page: 957 Line: 32341)

2858    In the RATIONALE section, delete the paragraph:

2859    The *lstat()* function is not required to update the time-related fields if the named file is not a symbolic link.  
 2860    While the *st\_uid*, *st\_gid*, *st\_atim*, *st\_mtim*, and *st\_ctim* members of the **stat** structure may apply to a  
 2861    symbolic link, they are not required to do so. No functions in POSIX.1-2008 are required to maintain any  
 2862    of these time fields.

2863    *Rationale*: Austin Group Defect Report(s) applied: 889. See <http://austingroupbugs.net/view.php?id=889>.

2864    **Change Number: XSH/TC2/D4/0140** [591]

2865    On Page: 968 Line: 32362 Section: futimens()

2866    Before the utimensat SYNOPSIS line, insert a line with OH shading:

2867    `#include <fcntl.h>`

2868    *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

2869    **Change Number: XSH/TC2/D4/0141** [817]

2870    On Page: 968 Line: 32386 Section: futimens()  
 2871    (2013 edition Page: 976 Line: 32925)

2872    In the DESCRIPTION section, change from:

2873    If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
 2874    are permitted using the current permissions of the directory underlying the file descriptor. If the file  
 2875    descriptor was opened with O\_SEARCH, the function shall not perform the check.

2876    to:

2877    If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
 2878    function shall check whether directory searches are permitted using the current permissions of the directory  
 2879    underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

2880    *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

2881   **Change Number: XSH/TC2/D4/0142** [485]

2882   On Page: 969 Line: 32407-32408 Section: futimens()

2883   In the DESCRIPTION section, change from:

2884   Upon completion, *futimens()* and *utimensat()* shall mark the last file status change timestamp for update.

2885   to:

2886   Upon successful completion, *futimens()* and *utimensat()* shall mark the last file status change timestamp for update, with the exception that if both *tv\_nsec* fields are set to *UTIME OMIT*, the file status change timestamp need not be marked for update.

2887   *Rationale*: Austin Group Defect Report(s) applied: 485. See <http://austingroupbugs.net/view.php?id=485>.

2888   The standard requires that timestamps be marked for update even when values being written are identical to what is already present. However, it is unclear whether *utimensat* is required to write anything when both arguments use *UTIME OMIT*. The intent is that implementations may, but not must, optimize these two cases.

2894   **Change Number: XSH/TC2/D4/0143** [817]

2895   On Page: 969 Line: 32435 Section: futimens()

2896   (2013 edition Page: 977 Line: 32973)

2897   In the ERRORS section, for the [EACCES] error, change from:

2898    $fd$  was not opened with *O\_SEARCH* and ...

2899   to:

2900   The access mode of the open file description associated with  $fd$  is not *O\_SEARCH* and ...

2901   *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

2902   **Change Number: XSH/TC2/D4/0144** [73]

2903   On Page: 973 Line: 32586 Section: fwprintf()

2904   In the DESCRIPTION section, change from:

2905   In format wide-character strings containing the % form of conversion specification, each argument in the argument list shall be used exactly once.

2906   to:

2907   In format wide-character strings containing the % form of conversion specification, each argument in the argument list shall be used exactly once. It is unspecified whether an encoding error occurs if the format string contains *wchar\_t* values that do not correspond to members of the character set of the current locale and the specified semantics do not require that value to be processed by *wcrtomb()*.

2912 *Rationale:* Austin Group Defect Report(s) applied: 73. See <http://austingroupbugs.net/view.php?id=73>.  
 2913  
 2914 A clarification has been made in the C11 standard.

2915 **Change Number: XSH/TC2/D4/0145** [894]

2916 On Page: 974 Line: 32634 Section: fwprintf()  
 2917 (2013 edition Page: 983 Line: 33180)

2918 In the DESCRIPTION section, change from:

2919 For o conversion, it increases the precision (if necessary) to force the first digit of the result to be zero.

2920 to:

2921 For o conversion, it shall increase the precision, if and only if necessary, to force the first digit of the result  
 2922 to be a zero (if the value and precision are both 0, a single 0 is printed).

2923 *Rationale:* Austin Group Defect Report(s) applied: 894. See <http://austingroupbugs.net/view.php?id=894>.

2924 **Change Number: XSH/TC2/D4/0146** [557]

2925 On Page: 979 Line: 32817,32838 Section: fwprintf()

2926 In the ERRORS section, on line 32817 delete:

2927 In addition, all forms of fwprintf() may fail if:

2928 [EINVAL] There are insufficient arguments.

2929 In the RATIONALE section, on line 32838 change from:

2930 None

2931 to:

2932 If an implementation detects that there are insufficient arguments for the format, it is recommended that the  
 2933 function should fail and report an [EINVAL] error.

2934 *Rationale:* Austin Group Defect Report(s) applied: 557. See <http://austingroupbugs.net/view.php?id=557>.  
 2935 The descriptions of fprintf() and fwprintf() say "The results are undefined if there are insufficient  
 2936 arguments for the format." However, the ERRORS section on each of those pages also specifies a "may  
 2937 fail" EINVAL error for this condition.

2938 **Change Number: XSH/TC2/D4/0147** [936]

2939 On Page: 979 Line: 32843 Section: fwprintf()  
 2940 (2013 edition Page: 988 Line: 33389)

2941 In the SEE ALSO section, add <inttypes.h> to the list.

2942 *Rationale:* Austin Group Defect Report(s) applied: 936. See <http://austingroupbugs.net/view.php?id=936>.

2943 **Change Number: XSH/TC2/D4/0148** [73]

2944 On Page: 983 Line: 32957 Section: fwscanf()

2945 In the DESCRIPTION section, change from:

2946 The format is a wide-character string composed of zero or more directives. Each directive is composed of  
2947 one of the following: one or more white-space wide characters (<space>, <tab>, <newline>, <vertical-tab>  
2948 or <form-feed>); an ordinary wide character (neither '%' nor a white-space character); or a conversion  
2949 specification.

2950 to:

2951 The format is a wide-character string composed of zero or more directives. Each directive is composed of  
2952 one of the following: one or more white-space wide characters (<space>, <tab>, <newline>, <vertical-tab>  
2953 or <form-feed>); an ordinary wide character (neither '%' nor a white-space character); or a conversion  
2954 specification. It is unspecified whether an encoding error occurs if the format string contains **wchar\_t**  
2955 values that do not correspond to members of the character set of the current locale and the specified  
2956 semantics do not require that value to be processed by *wcrtomb()*.

2957 *Rationale:* Austin Group Defect Report(s) applied: 73. See <http://austingroupbugs.net/view.php?id=73>.  
2958 A clarification has been made in the C11 standard.

2959 **Change Number: XSH/TC2/D4/0149** [823]

2960 On Page: 988 Line: 33162 Section: fwscanf()  
2961 (2013 edition Page: 997 Line: 33711)

2962 In the RETURN VALUE section, change from:

2963 If the input ends before the first matching failure or conversion, EOF shall be returned. If any error occurs,  
2964 EOF shall be returned, [CX] and *errno* shall be set to indicate the error[/CX]. If a read error occurs, the  
2965 error indicator for the stream shall be set.

2966 to:

2967 If the input ends before the first conversion (if any) has completed, and without a matching failure having  
2968 occurred, EOF shall be returned. If an error occurs before the first conversion (if any) has completed, and  
2969 without a matching failure having occurred, EOF shall be returned [CX] and *errno* shall be set to indicate  
2970 the error[/CX]. If a read error occurs, the error indicator for the stream shall be set.

2971 *Rationale:* Austin Group Defect Report(s) applied: 823. See <http://austingroupbugs.net/view.php?id=823>.

2972 **Change Number: XSH/TC2/D4/0150** [936]

2973 On Page: 989 Line: 33200 Section: fwscanf()  
2974 (2013 edition Page: 998 Line: 33749)

2975 In the SEE ALSO section, add <inttypes.h> to the list.

2976 *Rationale*: Austin Group Defect Report(s) applied: 936. See <http://austingroupbugs.net/view.php?id=936>.

2977 **Change Number: XSH/TC2/D4/0151** [826]

2978 On Page: 993 Line: 33318 Section: getc\_unlocked()

2979 In the DESCRIPTION section change from:

2980 ... in a thread-safe manner. They may only safely be used within a scope protected by *flockfile()* (or  
 2981 *ftrylockfile()*) and *funlockfile()*. These functions may safely be used ...

2982 to:

2983 ... in a fully thread-safe manner. They shall be thread-safe when used within a scope protected by *flockfile()*  
 2984 (or *ftrylockfile()*) and *funlockfile()*. These functions can safely be used ...

2985 *Rationale*: Austin Group Defect Report(s) applied: 826. See <http://austingroupbugs.net/view.php?id=826>.  
 2986 The thread safety of these functions needs to be specified clearly.

2987 **Change Number: XSH/TC2/D4/0152** [796]

2988 On Page: 1003 Line: 33674 Section: getdate()  
 2989 (2013 edition Page: 1012 Line: 34234)

2990 Change from:

2991 is Mon Sep 22 12:19:47 EDT 1986 and the LC\_TIME category is set to the default C locale:

2992 to:

2993 is Mon Sep 22 12:19:47 EDT 1986 and the LC\_TIME category is set to the default C or POSIX locale:

2994 *Rationale*: Austin Group Defect Report(s) applied: 796. See <http://austingroupbugs.net/view.php?id=796>.

2995 **Change Number: XSH/TC2/D4/0153** [569]

2996 On Page: 1005 Line: 33740,33751 Section: getdelim()

2997 In the DESCRIPTION section, at line 33740 change from:

2998 The size of the object pointed to by *\*lineptr* shall be increased to fit the incoming line, if it isn't already  
 2999 large enough, including room for the delimiter and a terminating NUL. The characters read, including any  
 3000 delimiter, shall be stored in the string pointed to by the *lineptr* argument, and a terminating NUL added  
 3001 when the delimiter or end of file is encountered.

3002 to:

3003 If *\*lineptr* is a null pointer or if the object pointed to by *\*lineptr* is of insufficient size, an object shall be  
 3004 allocated as if by *malloc()* or the object shall be reallocated as if by *realloc()*, respectively, such that the  
 3005 object is large enough to hold the characters to be written to it, including the terminating NUL, and *\*n* shall  
 3006 be set to the new size. If the object was allocated, or if the reallocation operation moved the object, *\*lineptr*

3007 shall be updated to point to the new object or new location. The characters read, including any delimiter,  
3008 shall be stored in the object, and a terminating NUL added when the delimiter or end of file is encountered.

3009 In the RETURN VALUE section, at line 33751 change from:

3010 shall return the number of characters written

3011 to:

3012 shall return the number of bytes written

3013 *Rationale:* Austin Group Defect Report(s) applied: 569. See <http://austingroupbugs.net/view.php?id=569>.  
3014 The changes described here are made to match existing practice and describe what was originally intended.

3015 **Change Number: XSH/TC2/D4/0154 [571]**

3016 On Page: 1005 Line: 33753 Section: getdelim()

3017 In the RETURN VALUE section, change from:

3018 If no characters were read, and the end-of-file indicator for the stream is set, or if the stream is at end-of-  
3019 file, the end-of-file indicator for the stream shall be set and the function shall return -1.

3020 to:

3021 If the end-of-file indicator for the stream is set, or if no characters were read and the stream is at end-of-  
3022 file, the end-of-file indicator for the stream shall be set and the function shall return -1.

3023 *Rationale:* Austin Group Defect Report(s) applied: 571. See <http://austingroupbugs.net/view.php?id=571>.  
3024 The changes described here are made to match existing practice and describe what was originally intended.

3025 **Change Number: XSH/TC2/D4/0155 [570]**

3026 On Page: 1005 Line: 33764 Section: getdelim()

3027 In the ERRORS section, replace the EOVERRLOW description with:

3028 The number of bytes to be written into the buffer, including the delimiter character (if encountered), would  
3029 exceed {SSIZE\_MAX}.

3030 *Rationale:* Austin Group Defect Report(s) applied: 570. See <http://austingroupbugs.net/view.php?id=570>.  
3031 The changes described here are made to match existing practice and describe what was originally intended.

3032 **Change Number: XSH/TC2/D4/0156 [511]**

3033 On Page: 1007 Line: 33813,33824 Section: getegid()

3034 In the DESCRIPTION section, on line 33813 add a sentence:

3035 The *getegid()* function shall not modify *errno*.

3036 In the RATIONALE section, on line 33824 change from:

3037 None.

3038 to:

3039 In a conforming environment, *getegid()* will always succeed. It is possible for implementations to provide an extension where a process in a non-conforming environment will not be associated with a user or group ID. It is recommended that such implementations return **(gid\_t)-1** and set *errno* to indicate such an environment; doing so does not violate this standard, since such an environment is already an extension.

3043 *Rationale:* Austin Group Defect Report(s) applied: 511. See <http://austingroupbugs.net/view.php?id=511>.

3044 **Change Number: XSH/TC2/D4/0157** [656]

3045 On Page: 1008 Line: 33853 Section: *getenv()*

3046 In the DESCRIPTION section, change from:

3047 The string pointed to may be overwritten by a subsequent call to *getenv()*, [CX]*setenv()*, *unsetenv()*, [/CX] [XSI]or *putenv()*[/XSI] but shall not be overwritten by a call to any other function in this volume of POSIX.1-200x.

3050 to:

3051 [CX]The returned string pointer might be invalidated or[/CX] the string content might be overwritten by a subsequent call to *getenv()*, [CX]*setenv()*, *unsetenv()*, [/CX] [XSI]or *putenv()*[/XSI] but they shall not be affected by a call to any other function in this volume of POSIX.1-2008. [CX]The returned string pointer might also be invalidated if the calling thread is terminated.[/CX]

3055 *Rationale:* Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 3056 This item is a layered change on XSH/TC1/D5/0238 [75,428].

3057 The change is to add the following text to the end of the paragraph at 2013 edition P1017, L34414-34416:

3058 [CX]The returned string pointer might also be invalidated if the calling thread is terminated.[/CX]

3059 **Change Number: XSH/TC2/D4/0158** [511]

3060 On Page: 1011 Line: 33934,33945 Section: *geteuid()*

3061 In the DESCRIPTION section, on line 33934 add a sentence:

3062 The *geteuid()* function shall not modify *errno*.

3063 In the RATIONALE section, on line 33945 change from:

3064 None.

3065 to:

3066 In a conforming environment, *geteuid()* will always succeed. It is possible for implementations to provide  
3067 an extension where a process in a non-conforming environment will not be associated with a user or group  
3068 ID. It is recommended that such implementations return *(uid\_t)-1* and set *errno* to indicate such an  
3069 environment; doing so does not violate this standard, since such an environment is already an extension.

3070 *Rationale:* Austin Group Defect Report(s) applied: 511. See <http://austingroupbugs.net/view.php?id=511>.

3071 **Change Number: XSH/TC2/D4/0159** [511]

3072 On Page: 1012 Line: 33966,33977 Section: *getgid()*

3073 In the DESCRIPTION section, on line 33966 add a sentence

3074 The *getgid()* function shall not modify *errno*.

3075 In the RATIONALE section, on line 33977 change from:

3076 None.

3077 to:

3078 In a conforming environment, *getgid()* will always succeed. It is possible for implementations to provide an  
3079 extension where a process in a non-conforming environment will not be associated with a user or group ID.  
3080 It is recommended that such implementations return *(gid\_t)-1* and set *errno* to indicate such an  
3081 environment; doing so does not violate this standard, since such an environment is already an extension.

3082 *Rationale:* Austin Group Defect Report(s) applied: 511. See <http://austingroupbugs.net/view.php?id=511>.

3083 **Change Number: XSH/TC2/D4/0160** [808]

3084 On Page: 1014 Line: 34008 Section: *getgrgid()*

3085 In the DESCRIPTION add a new paragraph after the second paragraph:

3086 Applications wishing to check for error situations should set *errno* to 0 before calling *getgrgid()*. If  
3087 *getgrgid()* returns a null pointer and *errno* is set to non-zero, an error occurred.

3088 *Rationale:* Austin Group Defect Report(s) applied: 808. See <http://austingroupbugs.net/view.php?id=808>.

3089 **Change Number: XSH/TC2/D4/0161** [808]

3090 On Page: 1014 Line: 34019 Section: *getgrgid()*

3091 In the RETURN VALUE section, change from:

3092 On error, *errno* shall be set to indicate the error.

3093 to:

3094 If the requested entry was not found, *errno* shall not be changed. On error, *errno* shall be set to indicate the  
3095 error.

3096 *Rationale:* Austin Group Defect Report(s) applied: 808. See <http://austingroupbugs.net/view.php?id=808>.

3097 **Change Number: XSH/TC2/D4/0162** [656]

3098 On Page: 1014 Line: 34021 Section: getgrgid()

3099 In the RETURN VALUE section, change from:

3100 The return value may point to a static area which is overwritten by a subsequent call to *getrent()*,  
 3101 *getgrgid()*, or *getgrnam()*.

3102 to:

3103 The application shall not modify the structure to which the return value points, nor any storage areas  
 3104 pointed to by pointers within the structure. The returned pointer, and pointers within the structure, might be  
 3105 invalidated or the structure or the storage areas might be overwritten by a subsequent call to *getrent()*,  
 3106 *getgrgid()*, or *getgrnam()*. The returned pointer, and pointers within the structure, might also be invalidated  
 3107 if the calling thread is terminated.

3108 *Rationale:* Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 3109 This item is a layered change on XSH/TC1/D5/0241 [75].

3110 The change is to add the following text to the end of the paragraph at 2013 edition P1023, L34589-34592:

3111 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
 3112 terminated.

3113 **Change Number: XSH/TC2/D4/0163** [808]

3114 On Page: 1016 Line: 34082 Section: getgrgid()

3115 In the APPLICATION USAGE section, delete the first paragraph:

3116 Applications wishing to check for error situations should set *errno* to 0 before calling *getgrgid()*. If *errno* is  
 3117 set on return, an error occurred.

3118 *Rationale:* Austin Group Defect Report(s) applied: 808. See <http://austingroupbugs.net/view.php?id=808>.

3119 **Change Number: XSH/TC2/D4/0164** [808]

3120 On Page: 1018 Line: 34136 Section: getgrnam()

3121 In the DESCRIPTION section, add a new paragraph after the second paragraph:

3122 Applications wishing to check for error situations should set *errno* to 0 before calling *getgrnam()*. If  
 3123 *getgrnam()* returns a null pointer and *errno* is set to non-zero, an error occurred.

3124 *Rationale:* Austin Group Defect Report(s) applied: 808. See <http://austingroupbugs.net/view.php?id=808>.

3125

3126 **Change Number: XSH/TC2/D4/0165** [808]

3127 On Page: 1018 Line: 34147 Section: `getgrnam()`  
3128 (2013 edition Page: 1027 Line: 34718)

3129 In the RETURN VALUE section, change from:

3130 On error, *errno* shall be set to indicate the error.

3131 to:

3132 If the requested entry was not found, *errno* shall not be changed. On error, *errno* shall be set to indicate the  
3133 error.

3134 *Rationale*: Austin Group Defect Report(s) applied: 808. See <http://austingroupbugs.net/view.php?id=808>.

3135 **Change Number: XSH/TC2/D4/0166** [656]

3136 On Page: 1018 Line: 34149 Section: `getgrnam()`

3137 In the RETURN VALUE section, change from:

3138 The return value may point to a static area which is overwritten by a subsequent call to `getrent()`,  
3139 `getrgid()`, or `getgrnam()`.

3140 to:

3141 The application shall not modify the structure to which the return value points, nor any storage areas  
3142 pointed to by pointers within the structure. The returned pointer, and pointers within the structure, might be  
3143 invalidated or the structure or the storage areas might be overwritten by a subsequent call to `getrent()`,  
3144 `getrgid()`, or `getgrnam()`. The returned pointer, and pointers within the structure, might also be invalidated  
3145 if the calling thread is terminated.

3146 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
3147 This item is a layered change on XSH/TC1/D5/0242 [75].

3148 The change is to add the following text to the end of the paragraph at 2013 edition, P1027, L34720-34723:

3149 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
3150 terminated.

3151 **Change Number: XSH/TC2/D4/0167** [808]

3152 On Page: 1019 Line: 34196 Section: `getgrnam()`

3153 In the APPLICATION USAGE section, delete the first paragraph:

3154 Applications wishing to check for error situations should set *errno* to 0 before calling `getgrnam()`. If *errno*  
3155 is set on return, an error occurred.

3156 *Rationale*: Austin Group Defect Report(s) applied: 808. See <http://austingroupbugs.net/view.php?id=808>.

3157 Application developers requiring to check for errors from these functions should set `errno` to zero before  
 3158 calling the function if they need to determine if an error occurred.

3159 **Change Number: XSH/TC2/D4/0168** [656]

3160 On Page: 1029 Line: 34469 Section: `getlogin()`

3161 In the RETURN VALUE section, change from:

3162 The return value from `getlogin()` may point to static data whose content is overwritten by each call.

3163 to:

3164 The application shall not modify the string returned. The returned pointer might be invalidated or the string  
 3165 content might be overwritten by a subsequent call to `getlogin()`. The returned pointer and the string content  
 3166 might also be invalidated if the calling thread is terminated.

3167 *Rationale:* Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 3168 This item is a layered change on XSH/TC1/D5/0244 [75].

3169 The change is to add the following text to the end of the paragraph at 2013 edition, P1038, L35048-35049:

3170 The returned pointer and the string content might also be invalidated if the calling thread is terminated.

3171 **Change Number: XSH/TC2/D4/0169** [608]

3172 On Page: 1040 Line: 34848 Section: `getopt()`

3173 In the DESCRIPTION section change from:

3174 `getopt()` shall also print a diagnostic message to `stderr` in the format specified for the `getopts` utility.

3175 to:

3176 `getopt()` shall also print a diagnostic message to `stderr` in the format specified for the `getopts` utility, unless  
 3177 the `stderr` stream has wide orientation in which case the behavior is undefined.

3178 In the APPLICATION USAGE section, add a new paragraph after page 1043 line 34980:

3179 Applications which use wide character output functions with `stderr` should ensure that any calls to `getopt()`  
 3180 do not write to `stderr`, either by setting `opterr` to 0 or by ensuring the first character of `optstring` is always a  
 3181 <colon>.

3182 *Rationale:* Austin Group Defect Report(s) applied: 608. See <http://austingroupbugs.net/view.php?id=608>.

3183 **Change Number: XSH/TC2/D4/0170** [808]

3184 On Page: 1057 Line: 35318 Section: `getpwnam()`  
 3185 (2013 edition Page: 1065 Line: 35887)

3186 In the RETURN VALUE section, change from:

3187 On error, *errno* shall be set to indicate the error.

3188 to:

3189 If the requested entry was not found, *errno* shall not be changed. On error, *errno* shall be set to indicate the  
3190 error.

3191 *Rationale*: Austin Group Defect Report(s) applied: 808. See <http://austingroupbugs.net/view.php?id=808>.

3192 **Change Number: XSH/TC2/D4/0171** [656]

3193 On Page: 1057 Line: 35319 Section: *getpwnam()*

3194 In the RETURN VALUE section, change from:

3195 The return value may point to a static area which is overwritten by a subsequent call to *getpwent()*,  
3196 *getpwnam()*, or *getpwuid()*.

3197 to:

3198 The application shall not modify the structure to which the return value points, nor any storage areas  
3199 pointed to by pointers within the structure. The returned pointer, and pointers within the structure, might be  
3200 invalidated or the structure or the storage areas might be overwritten by a subsequent call to *getpwent()*,  
3201 *getpwnam()*, or *getpwuid()*. The returned pointer, and pointers within the structure, might also be  
3202 invalidated if the calling thread is terminated.

3203 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
3204 This item is a layered change on XSH/TC1/D5/0255 [75].

3205 The change is to add the following text to the end of the paragraph at 2013 edition, P1065, L35888-35891:

3206 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
3207 terminated.

3208 **Change Number: XSH/TC2/D4/0172** [808]

3209 On Page: 1061 Line: 35451 Section: *getpwuid()*  
3210 (2013 edition Page: 1069 Line: 36023)

3211 In the RETURN VALUE section, change from:

3212 On error, *errno* shall be set to indicate the error.

3213 to:

3214 If the requested entry was not found, *errno* shall not be changed. On error, *errno* shall be set to indicate the  
3215 error.

3216 *Rationale*: Austin Group Defect Report(s) applied: 808. See <http://austingroupbugs.net/view.php?id=808>.

3217

3218 **Change Number: XSH/TC2/D4/0173** [656]

3219 On Page: 1061 Line: 35452 Section: getpwuid()

3220 In the RETURN VALUE section, change from:

3221 The return value may point to a static area which is overwritten by a subsequent call to *getpwent()*,  
 3222 *getpwnam()*, or *getpwuid()*.

3223 to:

3224 The application shall not modify the structure to which the return value points, nor any storage areas  
 3225 pointed to by pointers within the structure. The returned pointer, and pointers within the structure, might be  
 3226 invalidated or the structure or the storage areas might be overwritten by a subsequent call to *getpwent()*,  
 3227 *getpwnam()*, or *getpwuid()*. The returned pointer, and pointers within the structure, might also be  
 3228 invalidated if the calling thread is terminated.

3229 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 3230 This item is a layered change on XSH/TC1/D5/0256 [75].

3231 The change is to add the following text to the end of the paragraph at 2013 edition, P1069, L36024-36027:

3232 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
 3233 terminated.

3234 **Change Number: XSH/TC2/D4/0174** [791]

3235 On Page: 1078 Line: 35928 Section: getsubopt()

3236 In the SYNOPSIS section, add a CX margin marker and shading to all the synopsis.

3237 *Rationale*: Austin Group Defect Report(s) applied: 791. See <http://austingroupbugs.net/view.php?id=791>.  
 3238 This function in <stdlib.h> is an extension over the C standard contents for the header.

3239 **Change Number: XSH/TC2/D4/0175** [511]

3240 On Page: 1083 Line: 36116 Section: getuid()

3241 In the DESCRIPTION section add a sentence:

3242 The *getuid()* function shall not modify *errno*.

3243 In the RATIONALE section on line 36134, change from:

3244 None.

3245 to:

3246 In a conforming environment, *getuid()* will always succeed. It is possible for implementations to provide an  
 3247 extension where a process in a non-conforming environment will not be associated with a user or group ID.  
 3248 It is recommended that such implementations return **(uid\_t)-1** and set *errno* to indicate such an

3249 environment; doing so does not violate this standard, since such an environment is already an extension.

3250 *Rationale*: Austin Group Defect Report(s) applied: 511. See <http://austingroupbugs.net/view.php?id=511>.

3251 **Change Number: XSH/TC2/D4/0176 [897]**

3252 On Page: 1083 Line: 36124 Section: getuid()  
3253 (2013 edition Page: 1091 Line: 36709)

3254 In the EXAMPLES section, change from:

3255 The following example sets the effective user ID and the real user ID of the current process to the real user  
3256 ID of the caller.

3257 `#include <unistd.h>`  
3258 `#include <sys/types.h>`  
3259 `...`  
3260 `setreuid(getuid(), getuid());`

3261 to:

3262 The following example sets the effective user ID of the calling process to the real user ID.

3263 `#include <unistd.h>`  
3264 `...`  
3265 `seteuid(getuid());`

3266 *Rationale*: Austin Group Defect Report(s) applied: 897. See <http://austingroupbugs.net/view.php?id=897>.

3267 **Change Number: XSH/TC2/D4/0177 [506]**

3268 On Page: 1093 Line: 36468 Section: grantpt()

3269 In the ERRORS section, move line 36472 prior to 36470 (sort [EACCES] before [EBADF]).

3270 *Rationale*: Austin Group Defect Report(s) applied: 506. See <http://austingroupbugs.net/view.php?id=506>.  
3271 This is an editorial change.

3272 **Change Number: XSH/TC2/D4/0178 [777]**

3273 On Page: 1116 Line: 37199 Section: inet\_ntop()

3274 In the DESCRIPTION section, change from:

3275 The preferred form is "x:x:x:x:x:x:x:x", where the 'x's are the hexadecimal values of the eight 16-bit  
3276 pieces of the address. Leading zeros in individual fields can be omitted, but there shall be at least one  
3277 numeral in every field.

3278 to:

3279 The preferred form is "x:x:x:x:x:x:x:x", where the 'x's are the hexadecimal values of the eight 16-bit  
3280 pieces of the address. Leading zeros in individual fields can be omitted, but there shall be one to four

3281    hexadecimal digits in every field.

3282    *Rationale*: Austin Group Defect Report(s) applied: 777. See <http://austingroupbugs.net/view.php?id=777>.

3283    **Change Number: XSH/TC2/D4/0179** [743]

3284    On Page: 1119 Line: 37298 Section: initstate()  
3285    (2013 edition Page: 1128 Line: 37891)

3286    In the APPLICATION USAGE section, add a new paragraph to the end of the section:

3287    These functions should be avoided whenever non-trivial requirements (including safety) have to be  
3288    fulfilled.

3289    *Rationale*: Austin Group Defect Report(s) applied: 743. See <http://austingroupbugs.net/view.php?id=743>.

3290    **Change Number: XSH/TC2/D4/0180** [685]

3291    On Page: 1169 Line: 38878 Section: iswalnum()

3292    In the DESCRIPTION section, change from:

3293    is a wide-character code corresponding to a valid character in the current locale

3294    to:

3295    is a wide-character code corresponding to a valid character in the locale used by the function

3296    *Rationale*: Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3297    **Change Number: XSH/TC2/D4/0181** [685]

3298    On Page: 1171 Line: 38926 Section: iswalpha()

3299    In the DESCRIPTION section, change from:

3300    is a wide-character code corresponding to a valid character in the current locale

3301    to:

3302    is a wide-character code corresponding to a valid character in the locale used by the function

3303    *Rationale*: Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3304    **Change Number: XSH/TC2/D4/0182** [685]

3305    On Page: 1173 Line: 38974 Section: iswblank()

3306    In the DESCRIPTION section, change from:

3307 is a wide-character code corresponding to a valid character in the current locale

3308 to:

3309 is a wide-character code corresponding to a valid character in the locale used by the function

3310 *Rationale:* Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3311 **Change Number: XSH/TC2/D4/0183** [685]

3312 On Page: 1174 Line: 39015 Section: iswcntrl()

3313 In the DESCRIPTION section, change from:

3314 is a wide-character code corresponding to a valid character in the current locale

3315 to:

3316 is a wide-character code corresponding to a valid character in the locale used by the function

3317 *Rationale:* Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3318 **Change Number: XSH/TC2/D4/0184** [799]

3319 On Page: 1176 Line: 39070 Section: iswctype()  
(2013 edition Page: 1188 Line: 39711)

3320 In the RETURN VALUE section, change from:

3321 If *charclass* is 0

3322 to:

3323 If *charclass* is **(wctype\_t)0**

3324 *Rationale:* Austin Group Defect Report(s) applied: 799. See <http://austingroupbugs.net/view.php?id=799>.

3325 **Change Number: XSH/TC2/D4/0185** [799]

3326 On Page: 1177 Line: 39130 Section: iswctype()  
(2013 edition Page: 1189 Line: 39770)

3327 In the CHANGE HISTORY section, change from:

3328 The behavior of *n*=0 is now described.

3329 to:

3330 The behavior of *charclass* = **(wctype\_t)0** is now described.

3333 *Rationale*: Austin Group Defect Report(s) applied: 799. See <http://austingroupbugs.net/view.php?id=799>.

3334 **Change Number: XSH/TC2/D4/0186** [685]

3335 On Page: 1178 Line: 39149 Section: iswdigit()

3336 In the DESCRIPTION section, change from:

3337 is a wide-character code corresponding to a valid character in the current locale

3338 to:

3339 is a wide-character code corresponding to a valid character in the locale used by the function

3340 *Rationale*: Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3341 **Change Number: XSH/TC2/D4/0187** [685]

3342 On Page: 1180 Line: 39197 Section: iswgraph()

3343 In the DESCRIPTION section, change from:

3344 is a wide-character code corresponding to a valid character in the current locale

3345 to:

3346 is a wide-character code corresponding to a valid character in the locale used by the function

3347 *Rationale*: Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3348 **Change Number: XSH/TC2/D4/0188** [685]

3349 On Page: 1182 Line: 39245 Section: iswlower()

3350 In the DESCRIPTION section, change from:

3351 is a wide-character code corresponding to a valid character in the current locale

3352 to:

3353 is a wide-character code corresponding to a valid character in the locale used by the function

3354 *Rationale*: Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3355 **Change Number: XSH/TC2/D4/0189** [685]

3356 On Page: 1184 Line: 39293 Section: iswprint()

3357 In the DESCRIPTION section, change from:

3358 is a wide-character code corresponding to a valid character in the current locale

3359 to:

3360 is a wide-character code corresponding to a valid character in the locale used by the function

3361 *Rationale:* Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3362 **Change Number: XSH/TC2/D4/0190** [685]

3363 On Page: 1186 Line: 39341 Section: iswpunct()

3364 In the DESCRIPTION section, change from:

3365 is a wide-character code corresponding to a valid character in the current locale

3366 to:

3367 is a wide-character code corresponding to a valid character in the locale used by the function

3368 *Rationale:* Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3369 **Change Number: XSH/TC2/D4/0191** [685]

3370 On Page: 1188 Line: 39389 Section: iswspace()

3371 In the DESCRIPTION section, change from:

3372 is a wide-character code corresponding to a valid character in the current locale

3373 to:

3374 is a wide-character code corresponding to a valid character in the locale used by the function

3375 *Rationale:* Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3376 **Change Number: XSH/TC2/D4/0192** [685]

3377 On Page: 1190 Line: 39437 Section: iswupper()

3378 In the DESCRIPTION section, change from:

3379 is a wide-character code corresponding to a valid character in the current locale

3380 to:

3381 is a wide-character code corresponding to a valid character in the locale used by the function

3382 *Rationale:* Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3383 **Change Number: XSH/TC2/D4/0193** [685]

3384 On Page: 1192 Line: 39485 Section: iswxdigit()

3385 In the DESCRIPTION section, change from:

3386 is a wide-character code corresponding to a valid character in the current locale

3387 to:

3388 is a wide-character code corresponding to a valid character in the locale used by the function

3389 *Rationale:* Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

3390 **Change Number: XSH/TC2/D4/0194** [765]

3391 On Page: 1201 Line: 39708 Section: kill()

3392 (2013 edition Page: 1213 Line: 40379)

3393 In the RATIONALE section, change from:

3394 Existing implementations vary on the result of a *kill()* with *pid* indicating an inactive process (a terminated  
 3395 process that has not been waited for by its parent). Some indicate success on such a call (subject to  
 3396 permission checking), while others give an error of [ESRCH]. Since the definition of process lifetime in  
 3397 this volume of POSIX.1-2008 covers inactive processes, the [ESRCH] error as described is inappropriate in  
 3398 this case. In particular, this means that an application cannot have a parent process check for termination of  
 3399 a particular child with *kill()*. (Usually this is done with the null signal; this can be done reliably with  
 3400 *waitpid()*.)

3401 to:

3402 Historical implementations varied on the result of a *kill()* with *pid* indicating a zombie process. Some  
 3403 indicated success on such a call (subject to permission checking), while others gave an error of [ESRCH].  
 3404 Since the definition of process lifetime in this standard covers zombie processes, the [ESRCH] error as  
 3405 described is inappropriate in this case and implementations that give this error do not conform. This means  
 3406 that an application cannot have a parent process check for termination of a particular child by sending it the  
 3407 null signal with *kill()*, but must instead use *waitpid()* or *waitid()*.

3408 *Rationale:* Austin Group Defect Report(s) applied: 765. See <http://austingroupbugs.net/view.php?id=765>.

3409 **Change Number: XSH/TC2/D4/0195** [873]

3410 On Page: 1216 Line: 40102 Section: link()

3411 (2013 edition Page: 1228 Line: 40780)

3412 In the NAME section, delete:

3413 relative to two directory file descriptors

3414 *Rationale:* Austin Group Defect Report(s) applied: 873. See <http://austingroupbugs.net/view.php?id=873>.

3415 **Change Number: XSH/TC2/D4/0196** [591]

3416 On Page: 1216 Line: 40106 Section: link()

3417 Before the linkat SYNOPSIS line, insert a line with OH shading:

3418 `#include <fcntl.h>`

3419 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

3420 **Change Number: XSH/TC2/D4/0197** [817]

3421 On Page: 1216 Line: 40126 Section: link()

3422 (2013 edition Page: 1228 Line: 40805)

3423 In the DESCRIPTION section, change from:

3424 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
 3425 are permitted using the current permissions of the directory underlying the file descriptor. If the file  
 3426 descriptor was opened with O\_SEARCH, the function shall not perform the check.

3427 to:

3428 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
 3429 function shall check whether directory searches are permitted using the current permissions of the directory  
 3430 underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

3431 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

3432 **Change Number: XSH/TC2/D4/0198** [822]

3433 On Page: 1217 Line: 40156 Section: link()

3434 (2013 edition Page: 1229 Line: 40836)

3435 In the RETURN VALUE section, add after the [ENOENT] error:

3436 [ENOENT] or [ENOTDIR]

3437 The *path1* argument names an existing non-directory file, and the *path2* argument contains at least  
 3438 one non-<slash> character and ends with one or more trailing <slash> characters. If *path2* without  
 3439 the trailing <slash> characters would name an existing file, an [ENOENT] error shall not occur.

3440 *Rationale*: Austin Group Defect Report(s) applied: 822. See <http://austingroupbugs.net/view.php?id=822>.

3441 **Change Number: XSH/TC2/D4/0199** [817]

3442 On Page: 1217 Line: 40170 Section: link()

3443 (2013 edition Page: 1229 Line: 40853)

3444 In the ERRORS section, add:

3445 [EACCES]

3446 The access mode of the open file description associated with *fd1* or *fd2* is not O\_SEARCH and the  
 3447 permissions of the directory underlying *fd1* or *fd2* respectively do not permit directory searches.

3448 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

3449 **Change Number: XSH/TC2/D4/0200** [656]

3450 On Page: 1235 Line: 40719 Section: localeconv()

3451 In the RETURN VALUE section, change from:

3452 The application shall not modify the structure pointed to by the return value which may be overwritten by a  
 3453 subsequent call to *localeconv()*. In addition, calls to *setlocale()* with the categories LC\_ALL,  
 3454 LC\_MONETARY, or LC\_NUMERIC or calls to *use locale()* which change the categories  
 3455 LC\_MONETARY or LC\_NUMERIC may overwrite the contents of the structure.

3456 to:

3457 The application shall not modify the structure to which the return value points. [CX]nor any storage areas  
 3458 pointed to by pointers within the structure. The returned pointer, and pointers within the structure, might be  
 3459 invalidated or/[CX] the structure [CX]or the storage areas/[CX] might be overwritten by a subsequent call  
 3460 to *localeconv()*. In addition, [CX]the returned pointer, and pointers within the structure, might be  
 3461 invalidated or/[CX] the structure [CX]or the storage areas/[CX] might be overwritten by subsequent calls to  
 3462 *setlocale()* with the categories LC\_ALL, LC\_MONETARY, or LC\_NUMERIC, [CX]or by calls to  
 3463 *use locale()* which change the categories LC\_MONETARY or LC\_NUMERIC. The returned pointer,  
 3464 pointers within the structure, the structure, and the storage areas, might also be invalidated if the calling  
 3465 thread is terminated.[/CX]

3466 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 3467 This item is a layered change on XSH/TC1/D5/0362 [75].

3468 The change is to add the following text to the end of the paragraph at 2013 edition P1247, L41402-41409:

3469 The returned pointer, pointers within the structure, the structure, and the storage areas, might also be  
 3470 invalidated if the calling thread is terminated.

3471 **Change Number: XSH/TC2/D4/0201** [664]

3472 On Page: 1238 Line: 40811 Section: localtime()

3473 In the DESCRIPTION section, change from:

3474 If *localtime\_r()* does not set *tzname*, it shall not set *daylight* and shall not set *timezone*.

3475 to:

3476 If *localtime\_r()* sets *tzname*, it shall also set *daylight* and *timezone*. If *localtime\_r()* does not set *tzname*, it  
 3477 shall not set *daylight* and shall not set *timezone*.

3478 *Rationale*: Austin Group Defect Report(s) applied: 664. See <http://austingroupbugs.net/view.php?id=664>.  
 3479 The change is made for consistent behavior.

3480 **Change Number: XSH/TC2/D4/0202 [516]**

3481 On Page: 1256 Line: 41378,41395 Section: longjmp()

3482 In the DESCRIPTION section, at line 41378 change from:

3483 As it bypasses the usual function call and return mechanisms, *longjmp()* shall execute correctly in contexts  
3484 of interrupts, signals, and any of their associated functions. However, if *longjmp()* is invoked from a nested  
3485 signal handler (that is, from a function invoked as a result of a signal raised during the handling of another  
3486 signal), the behavior is undefined.

3487 to:

3488 Although *longjmp()* is an async-signal-safe function, if it is invoked from a signal handler which  
3489 interrupted a non-async-signal-safe function or equivalent (such as the processing equivalent to *exit()*)  
3490 performed after a return from the initial call to *main()*), the behavior of any subsequent call to a non-async-  
3491 signal-safe function or equivalent is undefined.

3492 In the APPLICATION USAGE section, at line 41395 add a new paragraph:

3493 It is recommended that applications do not call *longjmp()* or *siglongjmp()* from signal handlers. To avoid  
3494 undefined behavior when calling these functions from a signal handler the application needs to ensure one  
3495 of the following two things:

- 3496 1. After the call to *longjmp()* or *siglongjmp()* the process only calls async-signal-safe functions and  
3497 does not return from the initial call to *main()*.
- 3498 2. Any signal whose handler calls *longjmp()* or *siglongjmp()* is blocked during **every** call to a non-  
3499 async-signal-safe function, and no such calls are made after returning from the initial call to  
3500 *main()*.

3501 *Rationale:* Austin Group Defect Report(s) applied: 516. See <http://austingroupbugs.net/view.php?id=516>.

3502 The restrictions on using *longjmp()* and *siglongjmp()* are more restrictive than they need to be on POSIX  
3503 systems. The loosened restrictions presented here do not break existing implementations and make it easier  
3504 for application writers to create portable applications.

3505 **Change Number: XSH/TC2/D4/0203 [526]**

3506 On Page: 1268 Line: 41713,41718 Section: malloc()

3507 In the DESCRIPTION section, on line 41713 change from:

3508 If the size of the space requested is 0, the behavior is implementation-defined: the value returned shall be  
3509 either a null pointer or a unique pointer.

3510 to:

3511 If the size of the space requested is 0, the behavior is implementation-defined: either a null pointer shall be  
3512 returned, or the behavior shall be as if the size were some nonzero value, except that the behavior is  
3513 undefined if the returned pointer is used to access an object.

3514 In the RETURN VALUE section, on line 41718 change from:

3515 either a null pointer or a unique pointer that can be successfully passed to *free()* shall be returned.

3516 to:

3517 either:

3518 • a null pointer shall be returned [CX]and *errno* may be set to an implementation-defined  
 3519 value[/CX], or  
 3520 • a pointer to the allocated space shall be returned. The application shall ensure that the pointer is  
 3521 not used to access an object.

3522 *Rationale*: Austin Group Defect Report(s) applied: 526. See <http://austingroupbugs.net/view.php?id=526>.

3523 **Change Number: XSH/TC2/D4/0204** [663,674]

3524 On Page: 1270 Line: 41775 Section: *mblen()*  
 3525 (2013 edition Page: 1282 Line: 42472)

3526 In the ERRORS section, change from:

3527 [EILSEQ]  
 3528 [XSI]An invalid character sequence is detected.[/XSI]

3529 to:

3530 [EILSEQ]  
 3531 [CX]An invalid character sequence is detected. In the POSIX locale an EILSEQ error cannot occur  
 3532 since all byte values are valid characters.[/CX]

3533 On Page: 1272 Line: 41825 Section: *mbrlen()*  
 3534 (2013 edition Page: 1284 Line: 42526)

3535 In the ERRORS section, change from:

3536 [EILSEQ]  
 3537 An invalid character sequence is detected.

3538 to:

3539 [EILSEQ]  
 3540 An invalid character sequence is detected. [CX]In the POSIX locale an EILSEQ error cannot occur  
 3541 since all byte values are valid characters.[/CX]

3542 On Page: 1275 Line: 41890 Section: *mbrtowc()*  
 3543 (2013 edition Page: 1287 Line: 42594)

3544 In the ERRORS section, change from:

3545 [EILSEQ]  
 3546 An invalid character sequence is detected.

3547 to:

3548 [EILSEQ]

3549 An invalid character sequence is detected. [CX]In the POSIX locale an EILSEQ error cannot  
3550 occur since all byte values are valid characters.[/CX]

3551 *Rationale*: Austin Group Defect Report(s) applied: 663,674. See

3552 <http://austingroupbugs.net/view.php?id=663>.

3553 See <http://austingroupbugs.net/view.php?id=674>.

3554 **Change Number: XSH/TC2/D4/0205** [601]

3555 On Page: 1277 Line: 41984 Section: mbsrtowcs()

3556 In the DESCRIPTION section, change from:

3557 except that the conversion of characters pointed to by *src* is limited to at most *nmc* bytes (the size of the  
3558 input buffer).

3559 to (all within the CX shading):

3560 except that the conversion of characters indirectly pointed to by *src* is limited to at most *nmc* bytes (the size  
3561 of the input buffer), and under conditions where *mbsrtowcs()* would assign the address just past the last  
3562 character converted (if any) to the pointer object pointed to by *src*, *mbsnrtowcs()* shall instead assign the  
3563 address just past the last byte processed (if any) to that pointer object. If the input buffer ends with an  
3564 incomplete character, it is unspecified whether conversion stops at the end of the previous character (if  
3565 any), or at the end of the input buffer. In the latter case, a subsequent call to *mbsnrtowcs()* with an input  
3566 buffer that starts with the remainder of the incomplete character shall correctly complete the conversion of  
3567 that character.

3568 *Rationale*: Austin Group Defect Report(s) applied: 601. See <http://austingroupbugs.net/view.php?id=601>.

3569 **Change Number: XSH/TC2/D4/0206** [663]

3570 On Page: 1278 Line: 41998 Section: mbsrtowcs()

3571 (2013 edition Page: 1290 Line: 42706)

3572 In the ERRORS section, change from:

3573 [EILSEQ]

3574 An invalid character sequence is detected.

3575 to:

3576 [EILSEQ]

3577 An invalid character sequence is detected. [CX]In the POSIX locale an EILSEQ error cannot  
3578 occur since all byte values are valid characters.[/CX]

3579 *Rationale*: Austin Group Defect Report(s) applied: 663. See <http://austingroupbugs.net/view.php?id=663>.

3580 **Change Number: XSH/TC2/D4/0207** [601]

3581 On Page: 1278 Line: 42008 Section: mbsrtowcs()

3582 In the FUTURE DIRECTIONS section, change from:

3583 None.

3584 to:

3585 A future version may require that when the input buffer ends with an incomplete character, conversion  
 3586 stops at the end of the input buffer.

3587 *Rationale:* Austin Group Defect Report(s) applied: 601. See <http://austingroupbugs.net/view.php?id=601>.

3588 **Change Number: XSH/TC2/D4/0208** [663,674]

3589 On Page: 1279 Line: 42051 Section: mbstowcs()  
 3590 (2013 edition Page: 1291 Line: 42760)

3591 In the ERRORS section, change from:

3592 [EILSEQ]  
 3593 [XSI]An invalid byte sequence is detected.[/XSI]

3594 to:

3595 [EILSEQ]  
 3596 [CX]An invalid character sequence is detected. In the POSIX locale an EILSEQ error cannot  
 3597 occur since all byte values are valid characters.[/CX]

3598 *Rationale:* Austin Group Defect Report(s) applied: 663,674. See  
 3599 <http://austingroupbugs.net/view.php?id=663>.  
 3600 See <http://austingroupbugs.net/view.php?id=674>.

3601 **Change Number: XSH/TC2/D4/0209** [663,674]

3602 On Page: 1281 Line: 42104 Section: mbtowc()  
 3603 (2013 edition Page: 1293 Line: 42815)

3604 In the ERRORS section, change from:

3605 [EILSEQ]  
 3606 [XSI]An invalid character sequence is detected.[/XSI]

3607 to:

3608 [EILSEQ]  
 3609 [CX]An invalid character sequence is detected. In the POSIX locale an EILSEQ error cannot  
 3610 occur since all byte values are valid characters.[/CX]

3611 *Rationale:* Austin Group Defect Report(s) applied: 663,674. See  
 3612 <http://austingroupbugs.net/view.php?id=663>.

3613 See <http://austingroupbugs.net/view.php?id=674>.

3614 **Change Number: XSH/TC2/D4/0210** [873]

3615 On Page: 1289 Line: 42299 Section: mkdir()  
3616 (2013 edition Page: 1301 Line: 43016)

3617 In the NAME section, delete:

3618 relative to directory file descriptor

3619 *Rationale:* Austin Group Defect Report(s) applied: 873. See <http://austingroupbugs.net/view.php?id=873>.

3620 **Change Number: XSH/TC2/D4/0211** [591]

3621 On Page: 1289 Line: 42301 Section: mkdir()

3622 Before the mkdirat SYNOPSIS line, insert a line with OH shading:

3623 `#include <fcntl.h>`

3624 *Rationale:* Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

3625 **Change Number: XSH/TC2/D4/0212** [817]

3626 On Page: 1289 Line: 42323 Section: mkdir()  
3627 (2013 edition Page: 1301 Line: 43040)

3628 In the DESCRIPTION section, change from:

3629 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
3630 are permitted using the current permissions of the directory underlying the file descriptor. If the file  
3631 descriptor was opened with O\_SEARCH, the function shall not perform the check.

3632 to:

3633 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
3634 function shall check whether directory searches are permitted using the current permissions of the directory  
3635 underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

3636 *Rationale:* Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

3637 **Change Number: XSH/TC2/D4/0213** [817]

3638 On Page: 1290 Line: 42348 Section: mkdir()  
3639 (2013 edition Page: 1302 Line: 43066)

3640 In the ERRORS section, add:

3641 [EACCES]  
3642 The access mode of the open file description associated with *fd* is not O\_SEARCH and the

3643 permissions of the directory underlying *fd* do not permit directory searches.

3644 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

3645 **Change Number: XSH/TC2/D4/0214** [591]

3646 On Page: 1291 Line: 42393 Section: mkdir()

3647 Add a reference to <fcntl.h> to the SEE ALSO list.

3648 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

3649 **Change Number: XSH/TC2/D4/0215** [567,669]

3650 On Page: 1292 Line: 42419-42424 Section: mkdtemp()

3651 Replace the entire DESCRIPTION section with:

3652 **DESCRIPTION**

3653 The *mkdtemp()* function shall create a directory with a unique name derived from *template*.  
 3654 The application shall ensure that the string provided in *template* is a pathname ending with at  
 3655 least six trailing 'X' characters. The *mkdtemp()* function shall modify the contents of *template*  
 3656 by replacing six or more 'X' characters at the end of the pathname with the same number of  
 3657 characters from the portable filename character set. The characters shall be chosen such that  
 3658 the resulting pathname does not duplicate the name of an existing file at the time of the call  
 3659 to *mkdtemp()*. The *mkdtemp()* function shall use the resulting pathname to create the new  
 3660 directory as if by a call to:  
 3661  
 3662 `mkdir(pathname, S_IRWXU)`  
 3663  
 3664 The *mkstemp()* function shall create a regular file with a unique name derived from *template*  
 3665 and return a file descriptor for the file open for reading and writing. The application shall  
 3666 ensure that the string provided in *template* is a pathname ending with at least six trailing 'X'  
 3667 characters. The *mkstemp()* function shall modify the contents of *template* by replacing six or  
 3668 more 'X' characters at the end of the pathname with the same number of characters from the  
 3669 portable filename character set. The characters shall be chosen such that the resulting  
 3670 pathname does not duplicate the name of an existing file at the time of the call to *mkstemp()*.  
 3671 The *mkstemp()* function shall use the resulting pathname to create the file, and obtain a file  
 3672 descriptor for it, as if by a call to:  
 3673  
 3674 `open(filename, O_RDWR|O_CREAT|O_EXCL, S_IRUSR|S_IWUSR)`  
 3675  
 3676 By behaving as if the O\_EXCL flag for *open()* is set, the function prevents any possible race  
 3677 condition between testing whether the file exists and opening it for use.

3678 In the RETURN VALUE section, at line 42435 change from:

3679 Upon successful completion, the *mkdtemp()* function shall return a pointer to the string containing the  
 3680 directory name if it was created.

3681 to:

3682 Upon successful completion, the *mkdtemp()* function shall return the value of *template*.

3683 In the RETURN VALUE section, at line 42439 change from:

3684 Otherwise, it shall return -1 if no suitable file could be created.

3685 to:

3686 Otherwise, it shall return -1 and shall set *errno* to indicate the error.

3687 In the APPLICATION USAGE section, at line 42476 change from:

3688 The *mkdtemp()* and *mkstemp()* functions need not check to determine whether the filename part of *template* exceeds the maximum allowable filename length.

3690 to:

3691 Portable applications should pass exactly six trailing 'X's in the template and no more; implementations  
3692 may treat any additional trailing 'X's as either a fixed or replaceable part of the template. To be sure of only  
3693 passing six, a fixed string of at least one non-'X' character should precede the six 'X's.

3694

3695 Since 'X' is in the portable filename character set, some of the replacement characters can be 'X's, leaving  
3696 part (or even all) of the template effectively unchanged.

3697 *Rationale*: Austin Group Defect Report(s) applied: 567,669. See  
3698 <http://austingroupbugs.net/view.php?id=567>.  
3699 See <http://austingroupbugs.net/view.php?id=669>.

3700 **Change Number: XSH/TC2/D4/0216** [873]

3701 On Page: 1295 Line: 42496 Section: *mkfifo()*  
3702 (2013 edition Page: 1307 Line: 43219)

3703 In the NAME section, delete:

3704 relative to directory file descriptor

3705 *Rationale*: Austin Group Defect Report(s) applied: 873. See <http://austingroupbugs.net/view.php?id=873>.

3706 **Change Number: XSH/TC2/D4/0217** [591]

3707 On Page: 1295 Line: 42498 Section: *mkfifo()*

3708 Before the *mkfifoat* SYNOPSIS line, insert a line with OH shading:

3709 `#include <fcntl.h>`

3710 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

3711

3712 **Change Number: XSH/TC2/D4/0218** [817]

3713 On Page: 1295 Line: 42519 Section: mkfifo()  
 3714 (2013 edition Page: 1307 Line: 43242)

3715 In the DESCRIPTION section, change from:

3716 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
 3717 are permitted using the current permissions of the directory underlying the file descriptor. If the file  
 3718 descriptor was opened with O\_SEARCH, the function shall not perform the check.

3719 to:

3720 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
 3721 function shall check whether directory searches are permitted using the current permissions of the directory  
 3722 underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

3723 *Rationale:* Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

3724 **Change Number: XSH/TC2/D4/0219** [822]

3725 On Page: 1296 Line: 42537 Section: mkfifo()

3726 In the ERRORS section, change from:

3727 [ENOENT]

3728 A component of the path prefix specified by *path* does not name an existing directory or *path*  
 3729 is an empty string.

3730 to:

3731 [ENOENT]

3732 A component of the path prefix of *path* does not name an existing file or *path* is an empty  
 3733 string.

3734 [ENOENT] or [ENOTDIR]

3735 The *path* argument contains at least one non-<slash> character and ends with one or more  
 3736 trailing <slash> characters. If *path* without the trailing <slash> characters would name an  
 3737 existing file, an [ENOENT] error shall not occur.

3738 *Rationale:* Austin Group Defect Report(s) applied: 822. See <http://austingroupbugs.net/view.php?id=822>.  
 3739 This change is layered upon XSH/TC1/D5/0384 [146,435], changing the last sentence.

3740 **Change Number: XSH/TC2/D4/0220** [817]

3741 On Page: 1296 Line: 42544 Section: mkfifo()  
 3742 (2013 edition Page: 1308 Line: 43272)

3743 In the ERRORS section, for the [EACCES] error, change from:

3744 *fd* was not opened with O\_SEARCH and ...

3745 to:

3746 The access mode of the open file description associated with *fd* is not O\_SEARCH and ...

3747 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

3748 **Change Number: XSH/TC2/D4/0221** [591]

3749 On Page: 1297 Line: 42594 Section: mkfifo()

3750 Add a reference to <fcntl.h> to the SEE ALSO list.

3751 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

3752 **Change Number: XSH/TC2/D4/0222** [591]

3753 On Page: 1298 Line: 42617 Section: mknod()

3754 Before the mknodat SYNOPSIS line, insert a line with OH XSI shading:

3755 `#include <fcntl.h>`

3756 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

3757 **Change Number: XSH/TC2/D4/0223** [817]

3758 On Page: 1299 Line: 42669 Section: mknod()

3759 (2013 edition Page: 1313 Line: 43434)

3760 In the DESCRIPTION section, change from:

3761 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches are permitted using the current permissions of the directory underlying the file descriptor. If the file descriptor was opened with O\_SEARCH, the function shall not perform the check.

3764 to:

3765 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the function shall check whether directory searches are permitted using the current permissions of the directory underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

3768 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

3769 **Change Number: XSH/TC2/D4/0224** [822]

3770 On Page: 1299 Line: 42689 Section: mknod()

3771 In the ERRORS section, change from:

3772 [ENOENT]

3773 A component of the path prefix specified by *path* does not name an existing directory or *path*

3774 is an empty string.

3775 to:

3776 [ENOENT]  
 3777 A component of the path prefix of *path* does not name an existing file or *path* is an empty  
 3778 string.

3779 [ENOENT] or [ENOTDIR]  
 3780 The path argument contains at least one non-<slash> character and ends with one or more  
 3781 trailing <slash> characters. If *path* without the trailing <slash> characters would name an  
 3782 existing file, an [ENOENT] error shall not occur.

3783 *Rationale*: Austin Group Defect Report(s) applied: 822. See <http://austingroupbugs.net/view.php?id=822>.  
 3784 This change is layered upon XSH/TC1/D5/0390 [146,435], changing the last sentence.

3785 **Change Number: XSH/TC2/D4/0225** [817]

3786 On Page: 1300 Line: 42699 Section: mknod()  
 3787 (2013 edition Page: 1312 Line: 43399)

3788 In the ERRORS section, for the [EACCES] error, change from:

3789 *fd* was not opened with O\_SEARCH and ...

3790 to:

3791 The access mode of the open file description associated with *fd* is not O\_SEARCH and ...

3792 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

3793 **Change Number: XSH/TC2/D4/0226** [591]

3794 On Page: 1301 Line: 42745 Section: mknod()

3795 Add a reference to <font.h> to the SEE ALSO list.

3796 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

3797 **Change Number: XSH/TC2/D4/0227** [902]

3798 On Page: 1302 Line: 42761 Section: mkstemp()  
 3799 (2013 edition Page: 1315 Line: 43498)

3800 In the NAME section, change from:

3801 create a unique directory

3802 to:

3803 create a unique file

3804 *Rationale*: Austin Group Defect Report(s) applied: 902. See <http://austingroupbugs.net/view.php?id=902>.

3805 **Change Number: XSH/TC2/D4/0228** [724]

3806 On Page: 1303 Line: 42778-42792 Section: mktime()

3807 In the DESCRIPTION section, at line 42778 change from:

3808 The original values of the *tm\_wday* and *tm\_yday* components of the structure are ignored, and the original  
3809 values of the other components are not restricted to the ranges described in <**time.h**>.

3810 to:

3811 The original values of the *tm\_wday* and *tm\_yday* components of the structure shall be ignored, and the  
3812 original values of the other components shall not be restricted to the ranges described in <**time.h**>.

3813 At line 42789 change from:

3814 where the names in the structure and in the expression correspond

3815 to:

3816 where the names other than *tm\_yday* in the structure and in the expression correspond, and the *tm\_yday*  
3817 value used in the expression is the day of the year from 0 to 365 inclusive, calculated from the other **tm**  
3818 structure members specified in <**time.h**> (excluding *tm\_wday*).

3819 At line 42792 change from:

3820 and the other components are set to represent the specified time

3821 to:

3822 and the other components shall be set to represent the specified time

3823 *Rationale*: Austin Group Defect Report(s) applied: 724. See <http://austingroupbugs.net/view.php?id=724>.

3824 **Change Number: XSH/TC2/D4/0229** [852]

3825 On Page: 1311 Line: 43068 Section: mmap()  
3826 (2013 edition Page: 1324 Line: 43807)

3827 In the DESCRIPTION section, change from:

3828 If a MAP\_FIXED request is successful, the mapping established by *mmap()* replaces any previous  
3829 mappings for the pages in the range [*pa,pa+len*) of the process.

3830 to:

3831 If a MAP\_FIXED request is successful, then any previous mappings [ML|MLR] or memory  
3832 locks[ML|MLR] for those whole pages containing any part of the address range [*pa,pa+len*) shall be  
3833 removed, as if by an appropriate call to *munmap()*, before the new mapping is established.

3834 On Page: 1313 Line: 43189 Section: mmap()  
 3835 (2013 edition Page: 1326 Line: 43928)

3836 In the RATIONALE section, change from:

3837 If an application requests a mapping that would overlay existing mappings in the process, it might be  
 3838 desirable that an implementation detect this and inform the application. However, the default, portable (not  
 3839 MAP\_FIXED) operation does not overlay existing mappings. On the other hand, if the program specifies a  
 3840 fixed address mapping (which requires some implementation knowledge to determine a suitable address, if  
 3841 the function is supported at all), then the program is presumed to be successfully managing its own address  
 3842 space and should be trusted when it asks to map over existing data structures. Furthermore, it is also  
 3843 desirable to make as few system calls as possible, and it might be considered onerous to require an  
 3844 *munmap()* before an *mmap()* to the same address range. This volume of POSIX.1-2008 specifies that the  
 3845 new mappings replace any existing mappings, following existing practice in this regard.

3846 to:

3847 If an application requests a mapping that overlaps existing mappings in the process, it might be desirable  
 3848 that an implementation detect this and inform the application. However, if the program specifies a fixed  
 3849 address mapping (which requires some implementation knowledge to determine a suitable address, if the  
 3850 function is supported at all), then the program is presumed to be successfully managing its own address  
 3851 space and should be trusted when it asks to map over existing data structures. Furthermore, it is also  
 3852 desirable to make as few system calls as possible, and it might be considered onerous to require an  
 3853 *munmap()* before an *mmap()* to the same address range. This volume of POSIX.1-2008 specifies that the  
 3854 new mapping replaces any existing mappings (implying an automatic *munmap()* on the address range),  
 3855 following existing practice in this regard. The standard developers also considered whether there should be  
 3856 a way for new mappings to overlay existing mappings but found no existing practice for this.

3857 On Page: 1315 Line: 43244 Section: mmap()  
 3858 (2013 edition Page: 1328 Line: 43983)

3859 In the RATIONALE section, change from:

3860 and the MEMLOCK\_FUTURE argument

3861 to:

3862 and the MCL\_FUTURE argument

3863 *Rationale:* Austin Group Defect Report(s) applied: 852. See <http://austingroupbugs.net/view.php?id=852>.

3864 **Change Number: XSH/TC2/D4/0230** [504]

3865 On Page: 1339 Line: 44014 Section: mq\_unlink()

3866 In the ERRORS section, after line 44014 (EACCES) add:

3867 [EINTR]  
 3868       The call to *mq\_unlink()* blocked waiting for all references to the named message queue to be  
 3869       closed and a signal interrupted the call.

3870 *Rationale:* Austin Group Defect Report(s) applied: 504. See <http://austingroupbugs.net/view.php?id=504>.

3871 **Change Number: XSH/TC2/D4/0231** [909]

3872 On Page: 1360 Line: 44625 Section: nanosleep()  
3873 (2013 edition Page: 1373 Line: 45379)

3874 In the RETURN VALUE section, change from:

3875 The *rqtp* and *rmtlp* arguments may point to the same object.

3876 to:

3877 The *rqtp* and *rmtlp* arguments can point to the same object.

3878 *Rationale*: Austin Group Defect Report(s) applied: 909. See <http://austingroupbugs.net/view.php?id=909>.

3879 **Change Number: XSH/TC2/D4/0232** [781]

3880 On Page: 1364 Line: 44728 Section: newlocale()  
3881 (2013 edition Page: 1376 Line: 45474)

3882 In the DESCRIPTION section, change from:

3883 any of the other implementation-defined *LC\_\*\_MASK* values defined in <locale.h>

3884 to:

3885 any of the implementation-defined mask values defined in <locale.h>

3886 *Rationale*: Austin Group Defect Report(s) applied: 781. See <http://austingroupbugs.net/view.php?id=781>.

3887 **Change Number: XSH/TC2/D4/0233** [673]

3888 On Page: 1365 Line: 44763 Section: newlocale()

3889 In the EXAMPLES section, change from:

3890 and the *LC\_TIME* category data from a locale *tok2*

3891 to:

3892 and the *LC\_TIME* category data from a locale *loc2*

3893 *Rationale*: Austin Group Defect Report(s) applied: 673. See <http://austingroupbugs.net/view.php?id=673>.

3894 **Change Number: XSH/TC2/D4/0234** [656]

3895 On Page: 1375 Line: 45109 Section: nl\_langinfo()

3896 In the RETURN VALUE section, change from:

3897 This pointer may point to static data that may be overwritten on the next call to either function.

3898 to:

3899 The application shall not modify the string returned. The pointer returned by *nl\_langinfo()* might be  
 3900 invalidated or the string content might be overwritten by a subsequent call to *nl\_langinfo()* in any thread or  
 3901 to *nl\_langinfo\_l()* in the same thread or the initial thread, by subsequent calls to *setlocale()* with a category  
 3902 corresponding to the category of item (see <langinfo.h>) or the category LC\_ALL, or by subsequent calls  
 3903 to *use locale()* which change the category corresponding to the category of item. The pointer returned by  
 3904 *nl\_langinfo\_l()* might be invalidated or the string content might be overwritten by a subsequent call to  
 3905 *nl\_langinfo\_l()* in the same thread or to *nl\_langinfo()* in any thread, or by subsequent calls to *freelocale()* or  
 3906 *newlocale()* which free or modify the locale object that was passed to *nl\_langinfo\_l()*. The returned pointer  
 3907 and the string content might also be invalidated if the calling thread is terminated.

3908 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 3909 This item is a layered change on XSH/TC1/D5/0415 [75,402].

3910 The change is to add the following text to the end of the paragraph at 2013 edition P1387, L45872-45880:

3911 The returned pointer and the string content might also be invalidated if the calling thread is terminated.

3912 **Change Number: XSH/TC2/D4/0235** [873]

3913 On Page: 1379 Line: 45158 Section: *open()*  
 3914 (2013 edition Page: 1391 Line: 49540)

3915 In the NAME section, delete:

3916 relative to directory file descriptor

3917 *Rationale*: Austin Group Defect Report(s) applied: 873. See <http://austingroupbugs.net/view.php?id=873>.

3918 **Change Number: XSH/TC2/D4/0236** [835]

3919 On Page: 1379 Line: 45169 Section: *open()*  
 3920 (2013 edition Page: 1391 Line: 45951)

3921 In the DESCRIPTION section, change from:

3922 ... return a file descriptor for the named file that is the lowest file descriptor not currently open for that  
 3923 process.

3924 to:

3925 ... return a file descriptor for the named file, allocated as described in [xref to new section 2.14].

3926 *Rationale*: Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

3927

3928 **Change Number: XSH/TC2/D4/0237** [847]

3929 On Page: 1379 Line: 45193 Section: open()  
3930 (2013 edition Page: 1391 Line: 45975)

3931 In the DESCRIPTION section, change from:

3932 Otherwise, the file shall be created; ...

3933 to:

3934 Otherwise, if O\_DIRECTORY is not set the file shall be created as a regular file; ...

3935 On Page: 1381 Line: 45268 Section: open()  
3936 (2013 edition Page: 1393 Line: 46052)

3937 In the DESCRIPTION section, insert a new paragraph:

3938 If O\_CREAT and O\_DIRECTORY are set and the requested access mode is neither O\_WRONLY nor  
3939 O\_RDWR, the result is unspecified.

3940 *Rationale:* Austin Group Defect Report(s) applied: 847. See <http://austingroupbugs.net/view.php?id=847>.  
3941 The standard does not specify the behavior when open() is called with O\_CREAT|O\_RDONLY (or  
3942 O\_CREAT|O\_SEARCH) on an existing directory. Additionally, some systems want to allow creation of  
3943 directories using the open() function. This should be an allowed, but not required, extension.

3944 **Change Number: XSH/TC2/D4/0238** [817]

3945 On Page: 1381 Line: 45290 Section: open()  
3946 (2013 edition Page: 1394 Line: 46074)

3947 In the DESCRIPTION section, change from:

3948 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
3949 are permitted using the current permissions of the directory underlying the file descriptor. If the file  
3950 descriptor was opened with O\_SEARCH, the function shall not perform the check.

3951 to:

3952 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
3953 function shall check whether directory searches are permitted using the current permissions of the directory  
3954 underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

3955 *Rationale:* Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

3956 **Change Number: XSH/TC2/D4/0239** [835]

3957 On Page: 1382 Line: 45300 Section: open()  
3958 (2013 edition Page: 1394 Line: 46084)

3959 In the RETURN VALUE section, change from:

3960 representing the lowest numbered unused file descriptor

3961 to:

3962 representing the file descriptor

3963 *Rationale:* Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

3964 **Change Number: XSH/TC2/D4/0240** [847]

3965 On Page: 1382 Line: 45314 Section: open()  
 3966 (2013 edition Page: 1394 Line: 46098)

3967 In the ERRORS section, for the [EISDIR] error, change from:

3968 The named file is a directory and *oflag* includes O\_WRONLY or O\_RDWR.

3969 to:

3970 The named file is a directory and *oflag* includes O\_WRONLY or O\_RDWR, or includes O\_CREAT  
 3971 without O\_DIRECTORY.

3972 *Rationale:* Austin Group Defect Report(s) applied: 847. See <http://austingroupbugs.net/view.php?id=847>.  
 3973 The standard does not specify the behavior when open() is called with O\_CREAT|O\_RDONLY (or  
 3974 O\_CREAT|O\_SEARCH) on an existing directory. Additionally, some systems want to allow creation of  
 3975 directories using the open() function. This should be an allowed, but not required, extension.

3976 **Change Number: XSH/TC2/D4/0241** [822]

3977 On Page: 1382 Line: 45322 Section: open()

3978 In the ERRORS section, change from:

3979 [ENOENT]

3980 O\_CREAT is not set and the named file does not exist; or O\_CREAT is set and either the  
 3981 path prefix does not exist or the *path* argument points to an empty string.

3982 to:

3983 [ENOENT]

3984 O\_CREAT is not set and a component of path does not name an existing file, or O\_CREAT  
 3985 is set and a component of the path prefix of *path* does not name an existing file, or *path*  
 3986 points to an empty string.

3987 [ENOENT] or [ENOTDIR]

3988 O\_CREAT is set, and the path argument contains at least one non-<slash> character and ends  
 3989 with one or more trailing <slash> characters. If *path* without the trailing <slash> characters  
 3990 would name an existing file, an [ENOENT] error shall not occur.

3991 *Rationale:* Austin Group Defect Report(s) applied: 822. See <http://austingroupbugs.net/view.php?id=822>.  
 3992 This change is layered upon XSH/TC1/D5/0422 [146], changing the last sentence.

3993 **Change Number: XSH/TC2/D4/0242** [817]

3994 On Page: 1383 Line: 45345 Section: open()  
3995 (2013 edition Page: 1395 Line: 46134)

3996 In the ERRORS section, for the [EACCES] error, change from:

3997 *fd* was not opened with O\_SEARCH and ...

3998 to:

3999 The access mode of the open file description associated with *fd* is not O\_SEARCH and ...

4000 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

4001 **Change Number: XSH/TC2/D4/0243** [943]

4002 On Page: 1383 Line: 45361 Section: open()  
4003 (2013 edition Page: 1395 Line: 46152)

4004 In the ERRORS section, before the [ETXTBSY] error, add:

4005 [EOPNOTSUPP] The *path* argument names a socket.

4006 *Rationale*: Austin Group Defect Report(s) applied: 943. See <http://austingroupbugs.net/view.php?id=943>.

4007 **Change Number: XSH/TC2/D4/0244** [588]

4008 On Page: 1388 Line: 45564 Section: open\_memstream()

4009 In the DESCRIPTION section, add a new paragraph at the end of the section:

4010 After a successful *fclose()*, the pointer referenced by *bufp* can be passed to *free()*.

4011 *Rationale*: Austin Group Defect Report(s) applied: 588. See <http://austingroupbugs.net/view.php?id=588>.

4012 **Change Number: XSH/TC2/D4/0245** [586]

4013 On Page: 1388 Line: 45569 Section: open\_memstream()

4014 At the start of ERRORS section, insert:

4015 These functions shall fail if:

4016 [EMFILE]  
4017 {STREAM\_MAX} streams are currently open in the calling process.

4018 *Rationale*: Austin Group Defect Report(s) applied: 586. See <http://austingroupbugs.net/view.php?id=586>.

4019

4020 **Change Number: XSH/TC2/D4/0246** [632]

4021 On Page: 1396 Line: 45712 Section: pclose()

4022 At the end of the DESCRIPTION section, add a new paragraph:

4023 If a thread is canceled during execution of *pclose()*, the behavior is undefined.

4024 *Rationale*: Austin Group Defect Report(s) applied: 632. See <http://austingroupbugs.net/view.php?id=632>.

4025 **Change Number: XSH/TC2/D4/0247** [835]

4026 On Page: 1400 Line: 45837 Section: pipe()

4027 (2013 edition Page: 1413 Line: 46643)

4028 Change from:

4029 Their integer values shall be the two lowest available at the time of the *pipe()* call.

4030 to:

4031 The file descriptors shall be allocated as described in [xref to new section 2.14].

4032 *Rationale*: Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

4033 **Change Number: XSH/TC2/D4/0248** [467,835]

4034 On Page: 1400 Line: 45851 Section: pipe()

4035 (2013 edition Page: 1413 Line: 46657)

4036 Change from:

4037 ... otherwise, -1 shall be returned and *errno* set to indicate the error.

4038 to:

4039 ... otherwise, -1 shall be returned and *errno* set to indicate the error, no file descriptors shall be allocated and the contents of *fdes* shall be left unmodified.

4041 *Rationale*: Austin Group Defect Report(s) applied: 467,835. See <http://austingroupbugs.net/view.php?id=467>.

4042 See <http://austingroupbugs.net/view.php?id=835>.

4044 **Change Number: XSH/TC2/D4/0249** [623]

4045 On Page: 1404 Line: 45985 Section: poll()

4046 In the DESCRIPTION section, add a new paragraph to the end of the section:

4047 Provided the application does not perform any action that results in unspecified or undefined behavior, the value of the *fd* and *events* members of each element of *fds[]* shall not be modified by *poll()*.

4049 *Rationale*: Austin Group Defect Report(s) applied: 623. See <http://austingroupbugs.net/view.php?id=623>.

4050 **Change Number: XSH/TC2/D4/0250** [683]

4051 On Page: 1404 Line: 45987 Section: poll()  
4052 (2013 edition Page: 1417 Line: 46793)

4053 In the RETURN VALUE section, change from:

4054 Upon successful completion, *poll()* shall return a non-negative value. A positive value indicates the total  
4055 number of file descriptors that have been selected (that is, file descriptors for which the *revents* member is  
4056 non-zero).

4057 to:

4058 Upon successful completion, *poll()* shall return a non-negative value. A positive value indicates the total  
4059 number of **pollfd** structures that have selected events (that is, those for which the *revents* member is non-  
4060 zero).

4061 *Rationale*: Austin Group Defect Report(s) applied: 683. See <http://austingroupbugs.net/view.php?id=683>.

4062 **Change Number: XSH/TC2/D4/0251** [526]

4063 On Page: 1418 Line: 46423 Section: posix\_memalign()

4064 In the DESCRIPTION section change from:

4065 If the size of the space requested is 0, the behavior is implementation-defined; the value returned in *memptr*  
4066 shall be either a null pointer or a unique pointer.

4067 to:

4068 If the size of the space requested is 0, the behavior is implementation-defined: either a null pointer shall be  
4069 returned in *memptr*, or the behavior shall be as if the size were some nonzero value, except that the  
4070 behavior is undefined if the the value returned in *memptr* is used to access an object.

4071 *Rationale*: Austin Group Defect Report(s) applied: 526. See <http://austingroupbugs.net/view.php?id=526>.

4072 **Change Number: XSH/TC2/D4/0252** [520,526]

4073 On Page: 1418 Line: 46428,46436 Section: posix\_memalign()

4074 In the RETURN VALUE section, on line 46428 change from:

4075 otherwise, an error number shall be returned to indicate the error.

4076 to:

4077 otherwise, an error number shall be returned to indicate the error and the contents of *memptr* shall either be  
4078 left unmodified or be set to a null pointer.

4079 If *size* is 0, either:

4080     • *posix\_memalign()* shall not attempt to allocate any space, in which case either an implementation-defined error number shall be returned, or zero shall be returned with a null pointer returned in *memptr*

4081     • *posix\_memalign()* shall attempt to allocate some space and if the allocation succeeds, zero shall be returned and a pointer to the allocated space shall be returned in *memptr*. The application shall ensure that the pointer is not used to access an object.

4086 In the EXAMPLES section, on line 46436 change from:

4087 None

4088 to:

4089 The following example shows how applications can obtain consistent behavior on error by setting *\*memptr* to be a null pointer before calling *posix\_memalign()*.

```
4091 void *ptr = NULL;
4092 ...
4093 //do some work, which might goto error
4094 if (posix_memalign(&ptr, align, size))
4095     goto error;
4096
4097 //do some more work, which might goto error
4098 ...
4099 error:
4100     free(ptr);
4101     //more cleanup;
```

4102 *Rationale*: Austin Group Defect Report(s) applied: 520,526. See  
<http://austingroupbugs.net/view.php?id=520>.  
 4104 See <http://austingroupbugs.net/view.php?id=526>.

4105 **Change Number: XSH/TC2/D4/0253 [835]**

4106 On Page: 1420 Line: 46464 Section: *posix\_openpt()*  
 4107 (2013 edition Page: 1433 Line: 47270)

4108 Change from:

4109 The file descriptor is used by other I/O functions that refer to that pseudo-terminal.

4110 to:

4111 The file descriptor shall be allocated as described in [xref to new section 2.14] and can be used by other I/O  
 4112 functions that refer to that pseudo-terminal.

4113 *Rationale*: Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

4114

4115 **Change Number: XSH/TC2/D4/0254** [835]

4116 On Page: 1420 Line: 46474 Section: posix\_openpt()  
4117 (2013 edition Page: 1433 Line: 47281)

4118 Change from:

4119 ... shall open a master pseudo-terminal device and return a non-negative integer representing the lowest  
4120 numbered unused file descriptor.

4121 to:

4122 ... shall open a file descriptor for a master pseudo-terminal device and return a non-negative integer  
4123 representing the file descriptor.

4124 *Rationale:* Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

4125 **Change Number: XSH/TC2/D4/0255** [824]

4126 On Page: 1423 Line: 46571,46578 Section: posix\_spawn()

4127 In the DESCRIPTION section, at line 46571 change from:

4128 For those file descriptors that remain open, all attributes of the corresponding open file descriptions,  
4129 including file locks (see *fcntl()*), shall remain unchanged.

4130 to:

4131 For those file descriptors that remain open, the child process shall not inherit any file locks, but all  
4132 remaining attributes of the open file descriptions (see *fcntl()*) shall remain unchanged.

4133 At line 46578 (the second sentence in item 1 in the list), change from:

4134 All attributes of the corresponding open file descriptions, including file locks (see *fcntl()*), shall remain  
4135 unchanged.

4136 to:

4137 The child process shall not inherit any file locks, but all remaining attributes of the corresponding open file  
4138 descriptions (see *fcntl()*) shall remain unchanged.

4139 *Rationale:* Austin Group Defect Report(s) applied: 824. See <http://austingroupbugs.net/view.php?id=824>.  
4140 The behavior seen when creating a child using *posix\_spawn()* and when using *fork()*/*exec()* should be the  
4141 same.

4142 **Change Number: XSH/TC2/D4/0256** [835]

4143 On Page: 1515 Line: 48880 Section: posix\_typed\_mem\_open()  
4144 (2013 edition Page: 1528 Line: 49729)

4145 Change from:

4146 The file descriptor is used by other functions to refer to that typed memory object.

4147 to:

4148 The file descriptor shall be allocated as described in [xref to new section 2.14] and can be used by other  
 4149 functions to refer to that typed memory object.

4150 *Rationale:* Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

4151 **Change Number: XSH/TC2/D4/0257** [835]

4152 On Page: 1516 Line 48924 Section: posix\_typed\_mem\_open()  
 4153 (2013 edition Page: 1529 Line: 49773)

4154 Change from:

4155 ... return a file descriptor for the typed memory object that is the lowest numbered file descriptor not  
 4156 currently open for that process.

4157 to:

4158 ... return a file descriptor for the typed memory object.

4159 *Rationale:* Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

4160 **Change Number: XSH/TC2/D4/0258** [835]

4161 On Page: 1516 Line: 48939 Section: posix\_typed\_mem\_open()  
 4162 (2013 edition Page: 1529 Line: 49788)

4163 Change from:

4164 ... return a non-negative integer representing the lowest numbered unused file descriptor.

4165 to:

4166 ... return a non-negative integer representing the file descriptor.

4167 *Rationale:* Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

4168 **Change Number: XSH/TC2/D4/0259** [680]

4169 On Page: 1526 Line: 49239 Section: pselect()  
 4170 (2013 edition Page: 1539 Line: 50094)

4171 In the ERRORS section, change from:

4172 [EINTR]  
 4173 The function was interrupted before any of the selected events occurred and before the  
 4174 timeout interval expired.

4175 to:

4176 [EINTR]

4177 The function was interrupted while blocked waiting for any of the selected descriptors to  
4178 become ready and before the timeout interval expired.

4179 *Rationale*: Austin Group Defect Report(s) applied: 680. See <http://austingroupbugs.net/view.php?id=680>.

4180 **Change Number: XSH/TC2/D4/0260** [629]

4181 On Page: 1528 Line: 49313,49319 Section: psiginfo()

4182 In the NAME section, change from:

4183 print signal information to standard error

4184 to:

4185 write signal information to standard error

4186 In the DESCRIPTION section, at line 49319 change from:

4187 The *psiginfo()* and *psignal()* functions shall print a message out on *stderr* associated with a signal number.  
4188 If message is not null and is not the empty string, then the string pointed to by the *message* argument shall  
4189 be printed first, followed by a <colon>, a <space>, and the signal description string indicated by *signum*, or  
4190 by the signal associated with *pinfo*. If the *message* argument is null or points to an empty string, then only  
4191 the signal description shall be printed. For *psiginfo()*, the argument *pinfo* references a valid **siginfo\_t**  
4192 structure. For *psignal()*, if *signum* is not a valid signal number, the behavior is implementation-defined.

4193 to:

4194 The *psiginfo()* and *psignal()* functions shall write a language-dependent message associated with a signal  
4195 number to the standard error stream as follows:

4196     • First, if *message* is not a null pointer and is not the empty string, the string pointed to by the  
4197       *message* argument shall be written, followed by a <colon> and a <space>.

4198     • Then the signal description string associated with *signum* or with the signal indicated by *pinfo*  
4199       shall be written, followed by a <newline>.

4200 For *psiginfo()*, the application shall ensure that the argument *pinfo* references a valid **siginfo\_t** structure.  
4201 For *psignal()*, if *signum* is not a valid signal number, the behavior is implementation-defined.

4202 *Rationale*: Austin Group Defect Report(s) applied: 629. See <http://austingroupbugs.net/view.php?id=629>.

4203 **Change Number: XSH/TC2/D4/0261** [858]

4204 On Page: 1529 Line: 49363 Section: *pthread\_atfork()*  
4205 (2013 edition Page: 1543 Line: 50230)

4206 In the DESCRIPTION section, add a new paragraph:

4207 If a *fork()* call in a multi-threaded process leads to a *child* fork handler calling any function that is not  
 4208 async-signal-safe, the behavior is undefined.

4209 On Page: 1529 Line: 49377 Section: *pthread\_atfork()*  
 4210 (2013 edition Page: 1543 Line: 50244)

4211 In the APPLICATION USAGE section, change from:  
 4212 None.  
 4213 to:  
 4214 The original usage pattern envisaged for *pthread\_atfork()* was for the *prepare* fork handler to lock mutexes  
 4215 and other locks, and for the *parent* and *child* handlers to unlock them. However, since all of the relevant  
 4216 unlocking functions, except *sem\_post()*, are not async-signal-safe this usage results in undefined behavior  
 4217 in the child process unless the only such unlocking function it calls is *sem\_post()*.

4218 On Page: 1531 Line: 49441 Section: *pthread\_atfork()*  
 4219 (2013 edition Page: 1545 Line: 50308)

4220 In the FUTURE DIRECTIONS section, change from:  
 4221 None.  
 4222 to:  
 4223 The *pthread\_atfork()* function may be formally deprecated (for example by shading it OB) in a future  
 4224 revision of this standard.

4225 *Rationale:* Austin Group Defect Report(s) applied: 858. See <http://austingroupbugs.net/view.php?id=858>.

4226 **Change Number:** XSH/TC2/D4/0262 [757]

4227 On Page: 1541 Line: 49831 Section: *pthread\_attr\_getinheritsched()*  
 4228 (2013 edition Page: 1555 Line: 50699)

4229 In the CHANGE HISTORY section for Issue 7, change from:  
 4230 The *pthread\_attr\_getinheritsched()* and *pthread\_attr\_setinheritsched()* functions are moved from the  
 4231 Threads option.  
 4232 to:  
 4233 The *pthread\_attr\_getinheritsched()* and *pthread\_attr\_setinheritsched()* functions are marked only as part of  
 4234 the Thread Execution Scheduling option as the Threads option is now part of the Base.

4235 *Rationale:* Austin Group Defect Report(s) applied: 757. See <http://austingroupbugs.net/view.php?id=757>.

4236

4237 **Change Number: XSH/TC2/D4/0263 [757]**

4238 On Page: 1545 Line: 49961 Section: *pthread\_attr\_getschedpolicy()*  
4239 (2013 edition Page: 1559 Line: 50833)

4240 In the CHANGE HISTORY section for Issue 7, change from:

4241 The *pthread\_attr\_getschedpolicy()* and *pthread\_attr\_setschedpolicy()* functions are moved from the  
4242 Threads option.

4243 to:

4244 The *pthread\_attr\_getschedpolicy()* and *pthread\_attr\_setschedpolicy()* functions are marked only as part of  
4245 the Thread Execution Scheduling option as the Threads option is now part of the Base.

4246 *Rationale:* Austin Group Defect Report(s) applied: 757. See <http://austingroupbugs.net/view.php?id=757>.

4247 **Change Number: XSH/TC2/D4/0264 [757]**

4248 On Page: 1547 Line: 50023 Section: *pthread\_attr\_getscope()*  
4249 (2013 edition Page: 1561 Line: 50897)

4250 In the CHANGE HISTORY section for Issue 7, change from:

4251 The *pthread\_attr\_getscope()* and *pthread\_attr\_setscope()* functions are moved from the Threads option.

4252 to:

4253 The *pthread\_attr\_getscope()* and *pthread\_attr\_setscope()* functions are marked only as part of the Thread  
4254 Execution Scheduling option as the Threads option is now part of the Base.

4255 *Rationale:* Austin Group Defect Report(s) applied: 757. See <http://austingroupbugs.net/view.php?id=757>.

4256 **Change Number: XSH/TC2/D4/0265 [757]**

4257 On Page: 1552 Line: 50161 Section: *pthread\_attr\_getstacksize()*  
4258 (2013 edition Page: 1566 Line: 51036)

4259 In the CHANGE HISTORY section for Issue 7, change from:

4260 The *pthread\_attr\_getstacksize()* and *pthread\_attr\_setstacksize()* functions are moved from the Threads  
4261 option.

4262 to:

4263 The *pthread\_attr\_getstacksize()* and *pthread\_attr\_setstacksize()* functions are marked only as part of the  
4264 Thread Stack Size Attribute option as the Threads option is now part of the Base.

4265 *Rationale:* Austin Group Defect Report(s) applied: 757. See <http://austingroupbugs.net/view.php?id=757>.

4266

4267 **Change Number: XSH/TC2/D4/0266 [972]**

4268 On Page: 1562 Line: 50259-50261 Section: *pthread\_barrier\_destroy()*  
 4269 (2013 edition Page: 1576 Line: 51134-51136)

4270 In the DESCRIPTION section, change from:

4271 Only the object referenced by *barrier* may be used for performing synchronization. The result of referring  
 4272 to copies of that object in calls to *pthread\_barrier\_destroy()* or *pthread\_barrier\_wait()* is undefined.

4273 to:

4274 See [xref to section 2.9.9] for further requirements.

4275 On Page: 1568 Line: 50418-50421 Section: *pthread\_barrierattr\_getpshared()*  
 4276 (2013 edition Page: 1582 Line: 51293-51296)

4277 In the DESCRIPTION section, change from:

4278 If the *process-shared* attribute is PTHREAD\_PROCESS\_PRIVATE, the barrier shall only be operated  
 4279 upon by threads created within the same process as the thread that initialized the barrier; if threads of  
 4280 different processes attempt to operate on such a barrier, the behavior is undefined.

4281 to:

4282 See [xref to section 2.9.9] for further requirements.

4283 *Rationale:* Austin Group Defect Report(s) applied: 972. See <http://austingroupbugs.net/view.php?id=972>.

4284 **Change Number: XSH/TC2/D4/0267 [757]**

4285 On Page: 1569 Line: 50460 Section: *pthread\_barrierattr\_getpshared()*  
 4286 (2013 edition Page: 1583 Line: 51335)

4287 In the CHANGE HISTORY section for Issue 7, change from:

4288 The *pthread\_barrierattr\_getpshared()* and *pthread\_barrierattr\_setpshared()* functions are moved from the  
 4289 Barriers option.

4290 to:

4291 The *pthread\_barrierattr\_getpshared()* and *pthread\_barrierattr\_setpshared()* functions are marked only as  
 4292 part of the Thread Process-Shared Synchronization option as the Threads option is now part of the Base.

4293 *Rationale:* Austin Group Defect Report(s) applied: 757. See <http://austingroupbugs.net/view.php?id=757>.

4294 **Change Number: XSH/TC2/D4/0268 [624]**

4295 On Page: 1574 Line: 50552 Section: *pthread\_cleanup\_pop()*

4296 In the DESCRIPTION section, change from:

4297 These functions may be implemented as macros.

4298 to:

4299 It is unspecified whether *pthread\_cleanup\_push* and *pthread\_cleanup\_pop* are macros or functions. If a  
4300 macro definition is suppressed in order to access an actual function, or a program defines an external  
4301 identifier with any of these names the behavior is undefined.

4302 *Rationale*: Austin Group Defect Report(s) applied: 624. See <http://austingroupbugs.net/view.php?id=624>.

4303 **Change Number: XSH/TC2/D4/0269** [972]

4304 On Page: 1582 Line: 50872-50874 Section: *pthread\_cond\_destroy()*  
4305 (2013 edition Page: 1596 Line: 51751-51753)

4306 In the DESCRIPTION section, change from:

4307 Only *cond* itself may be used for performing synchronization. The result of referring to copies of *cond* in  
4308 calls to *pthread\_cond\_wait()*, *pthread\_cond\_timedwait()*, *pthread\_cond\_signal()*,  
4309 *pthread\_cond\_broadcast()*, and *pthread\_cond\_destroy()* is undefined.

4310 to:

4311 See [xref to section 2.9.9] for further requirements.

4312 *Rationale*: Austin Group Defect Report(s) applied: 972. See <http://austingroupbugs.net/view.php?id=972>.

4313 **Change Number: XSH/TC2/D4/0270** [910]

4314 On Page: 1583 Line: 50927 Section: *pthread\_cond\_destroy()*  
4315 (2013 edition Page: 1597 Line: 51806)

4316 In the EXAMPLES section, change from:

4317 `rp->notbusy`

4318 to:

4319 `ep->notbusy`

4320 *Rationale*: Austin Group Defect Report(s) applied: 910. See <http://austingroupbugs.net/view.php?id=910>.

4321 **Change Number: XSH/TC2/D4/0271** [749]

4322 On Page: 1587 Line: 51042 Section: *pthread\_cond\_timedwait()*

4323 In the RETURN VALUE section, change from:

4324 Except in the case of [ETIMEDOUT],

4325 to:

4326 Except for [ETIMEDOUT], [ENOTRECOVERABLE], and [EOWNERDEAD],

4327 *Rationale*: Austin Group Defect Report(s) applied: 749. See <http://austingroupbugs.net/view.php?id=749>.

4328 **Change Number: XSH/TC2/D4/0272** [972]

4329 On Page: 1596 Line: 51368-51371 Section: `pthread_condattr_getpshared()`  
 4330 (2013 edition Page: 1611 Line: 52252-52255)

4331 In the DESCRIPTION section, change from:

4332 If the *process-shared* attribute is PTHREAD\_PROCESS\_PRIVATE, the condition variable shall only be  
 4333 operated upon by threads created within the same process as the thread that initialized the condition  
 4334 variable; if threads of differing processes attempt to operate on such a condition variable, the behavior is  
 4335 undefined.

4336 to:

4337 See [xref to section 2.9.9] for further requirements.

4338 *Rationale*: Austin Group Defect Report(s) applied: 972. See <http://austingroupbugs.net/view.php?id=972>.

4339 **Change Number: XSH/TC2/D4/0273** [757]

4340 On Page: 1597 Line: 51409 Section: `pthread_condattr_getpshared()`  
 4341 (2013 edition Page: 1612 Line: 52293)

4342 In the CHANGE HISTORY section for Issue 7, change from:

4343 The `pthread_condattr_getpshared()` and `pthread_condattr_setpshared()` functions are moved from the  
 4344 Threads option.

4345 to:

4346 The `pthread_condattr_getpshared()` and `pthread_condattr_setpshared()` functions are marked only as part  
 4347 of the Thread Process-Shared Synchronization option as the Threads option is now part of the Base.

4348 *Rationale*: Austin Group Defect Report(s) applied: 757. See <http://austingroupbugs.net/view.php?id=757>.

4349 **Change Number: XSH/TC2/D4/0274** [849]

4350 On Page: 1602 Line: 51480 Section: `pthread_create()`

4351 In the APPLICATION USAGE section, change from:

4352 through the return value of the `pthread_create()` function

4353 to:

4354 through the *thread* argument of the `pthread_create()` function

4355 *Rationale*: Austin Group Defect Report(s) applied: 849. See <http://austingroupbugs.net/view.php?id=849>.

4356 **Change Number: XSH/TC2/D4/0275** [757]

4357 On Page: 1611 Line: 51797 Section: *pthread\_getcpu\_clockid()*  
4358 (2013 edition Page: 1626 Line: 52682)

4359 In the CHANGE HISTORY section for Issue 7, change from:

4360 The *pthread\_getcpu\_clockid()* function is moved from the Threads option.

4361 to:

4362 The *pthread\_getcpu\_clockid()* function is marked only as part of the Thread CPU-Time Clocks option as the  
4363 Threads option is now part of the Base.

4364 *Rationale*: Austin Group Defect Report(s) applied: 757. See <http://austingroupbugs.net/view.php?id=757>.

4365 **Change Number: XSH/TC2/D4/0276** [757]

4366 On Page: 1614 Line: 51885 Section: *pthread\_getschedparam()*  
4367 (2013 edition Page: 1629 Line: 52771)

4368 In the CHANGE HISTORY section for Issue 7, change from:

4369 The *pthread\_getschedparam()* and *pthread\_setschedparam()* functions are moved from the Threads option.

4370 to:

4371 The *pthread\_getschedparam()* and *pthread\_setschedparam()* functions are marked only as part of the  
4372 Thread Execution Scheduling option as the Threads option is now part of the Base.

4373 *Rationale*: Austin Group Defect Report(s) applied: 757. See <http://austingroupbugs.net/view.php?id=757>.

4374 **Change Number: XSH/TC2/D4/0277** [765]

4375 On Page: 1625 Line: 52287 Section: *pthread\_kill()*  
4376 (2013 edition Page: 1640 Line: 53174)

4377 In the RATIONALE section, add a paragraph:

4378 Existing implementations vary on the result of a *pthread\_kill()* with a thread ID indicating an inactive  
4379 thread (a terminated thread that has not been detached or joined). Some indicate success on such a call,  
4380 while others give an error of [ESRCH]. Since the definition of thread lifetime in this volume of POSIX.1-  
4381 2008 covers inactive threads, the [ESRCH] error as described is inappropriate in this case. In particular, this  
4382 means that an application cannot have one thread check for termination of another with *pthread\_kill()*.

4383 On Page: 1625 Line: 52289 Section: *pthread\_kill()*  
4384 (2013 edition Page: 1640 Line: 53176)

4385 In the FUTURE DIRECTIONS section, change from:

4386 None.

4387 to:

4388 A future version of this standard may require that *pthread\_kill()* not fail with ESRCH in the case of sending  
 4389 signals to an inactive thread (a terminated thread not yet detached or joined), even though no signal will be  
 4390 delivered because the thread is no longer running.

4391 *Rationale:* Austin Group Defect Report(s) applied: 765. See <http://austingroupbugs.net/view.php?id=765>.

4392 **Change Number: XSH/TC2/D4/0278** [811]

4393 On Page: 1628 Line: 52362 Section: *pthread\_mutex\_destroy()*  
 4394 (2013 edition Page: 1643 Line: 53249)

4395 In the DESCRIPTION section, change from:

4396 Attempting to destroy a locked mutex or a mutex that is referenced (for example, while being used in a  
 4397 *pthread\_cond\_timedwait()* or *pthread\_cond\_wait()*) by another thread results in undefined behavior.

4398 to:

4399 Attempting to destroy a locked mutex, or a mutex that another thread is attempting to lock, or a mutex that  
 4400 is being used in a *pthread\_cond\_timedwait()* or *pthread\_cond\_wait()* call by another thread, results in  
 4401 undefined behavior.

4402 *Rationale:* Austin Group Defect Report(s) applied: 811. See <http://austingroupbugs.net/view.php?id=811>.

4403 **Change Number: XSH/TC2/D4/0279** [972]

4404 On Page: 1628 Line: 52369-52371 Section: *pthread\_mutex\_destroy()*  
 4405 (2013 edition Page: 1643 Line: 53256-53258)

4406 In the DESCRIPTION section, change from:

4407 Only *mutex* itself may be used for performing synchronization. The result of referring to copies of *mutex* in  
 4408 calls to *pthread\_mutex\_lock()*, *pthread\_mutex\_trylock()*, *pthread\_mutex\_unlock()*, and  
 4409 *pthread\_mutex\_destroy()* is undefined.

4410 to:

4411 See [xref to section 2.9.9] for further requirements.

4412 *Rationale:* Austin Group Defect Report(s) applied: 972. See <http://austingroupbugs.net/view.php?id=972>.

4413 **Change Number: XSH/TC2/D4/0280** [811]

4414 On Page: 1632 Line: 52537-52557 Section: *pthread\_mutex\_destroy()*  
 4415 (2013 edition Page: 1647 Line: 53424-53444)

4416 In the RATIONALE section, change from:

4417 A mutex can be destroyed immediately after it is unlocked. For example, consider the following code:

```

4418 struct obj {
4419     pthread_mutex_t om;
4420     int refcnt;
4421     ...
4422 };
4423 obj_done(struct obj *op)
4424 {
4425     pthread_mutex_lock(&op->om);
4426     if (--op->refcnt == 0) {
4427         pthread_mutex_unlock(&op->om);
4428         (A)     pthread_mutex_destroy(&op->om);
4429         (B)     free(op);
4430     } else
4431         (C)     pthread_mutex_unlock(&op->om);
4432 }
```

4433 In this case *obj* is reference counted and *obj\_done()* is called whenever a reference to the object is dropped.  
 4434 Implementations are required to allow an object to be destroyed and freed and potentially unmapped (for  
 4435 example, lines A and B) immediately after the object is unlocked (line C).

4436 to:

4437 A mutex can be destroyed immediately after it is unlocked. However, since attempting to destroy a locked  
 4438 mutex, or a mutex that another thread is attempting to lock, or a mutex that is being used in a  
 4439 *pthread\_cond\_timedwait()* or *pthread\_cond\_wait()* call by another thread, results in undefined behavior,  
 4440 care must be taken to ensure that no other thread may be referencing the mutex.

4441 *Rationale*: Austin Group Defect Report(s) applied: 811. See <http://austingroupbugs.net/view.php?id=811>.

4442 **Change Number: XSH/TC2/D4/0281** [972]

4443 On Page: 1657 Line: 53355-53358 Section: *pthread\_mutexattr\_getpshared()*  
 4444 (2013 edition Page: 1672 Line: 54249-54252)

4445 In the DESCRIPTION section, change from:

4446 If the *process-shared* attribute is PTHREAD\_PROCESS\_PRIVATE, the mutex shall only be operated upon  
 4447 by threads created within the same process as the thread that initialized the mutex; if threads of differing  
 4448 processes attempt to operate on such a mutex, the behavior is undefined.

4449 to:

4450 See [xref to section 2.9.9] for further requirements.

4451 *Rationale*: Austin Group Defect Report(s) applied: 972. See <http://austingroupbugs.net/view.php?id=972>.

4452 **Change Number: XSH/TC2/D4/0282** [757]

4453 On Page: 1658 Line: 53396 Section: *pthread\_mutexattr\_getpshared()*  
 4454 (2013 edition Page: 1673 Line: 54290)

4455 In the CHANGE HISTORY section for Issue 7, change from:

4456 The *pthread\_mutexattr\_getpshared()* and *pthread\_mutexattr\_setpshared()* functions are moved from the  
 4457 Threads option.

4458 to:

4459 The *pthread\_mutexattr\_getpshared()* and *pthread\_mutexattr\_setpshared()* functions are marked only as  
 4460 part of the Thread Process-Shared Synchronization option as the Threads option is now part of the Base.

4461 *Rationale*: Austin Group Defect Report(s) applied: 757. See <http://austingroupbugs.net/view.php?id=757>.

4462 **Change Number: XSH/TC2/D4/0283** [748]

4463 On Page: 1659 Line: 53421 Section: *pthread\_mutexattr\_getrobust()*  
 4464 (2013 edition Page: 1744 Line: 54315)

4465 In the DESCRIPTION section, change from:

4466 ... the next thread that acquires the mutex may be notified about the termination by the return value  
 4467 [EOWNERDEAD]. The notified thread can then attempt to mark the state protected by the mutex as  
 4468 consistent again by a call to *pthread\_mutex\_consistent()*.

4469 to:

4470 ... the next thread that attempts to acquire the mutex may be notified about the termination by the return  
 4471 value [EOWNERDEAD]. The notified thread can then attempt to make the state protected by the mutex  
 4472 consistent again, and if successful can mark the mutex state as consistent by calling  
 4473 *pthread\_mutex\_consistent()*.

4474 *Rationale*: Austin Group Defect Report(s) applied: 748. See <http://austingroupbugs.net/view.php?id=748>.

4475 **Change Number: XSH/TC2/D4/0284** [863]

4476 On Page: 1669 Line: 53609 Section: *pthread\_once()*  
 4477 (2013 edition Page: 1684 Line: 54487)

4478 In the DESCRIPTION section, add new paragraph:

4479 If the call to *init\_routine* is terminated by a call to *longjmp()*, *\_longjmp()*, or *siglongjmp()*, the behavior is  
 4480 undefined.

4481 On Page: 1669 Line: 53621 Section: *pthread\_once()*  
 4482 (2013 edition Page: 1684 Line: 54499)

4483 In the APPLICATION USAGE section, change from:

4484 None.

4485 to:

4486 If *init\_routine* recursively calls *pthread\_once()* with the same *once\_control*, the recursive call will not call  
 4487 the specified *init\_routine*, and thus the specified *init\_routine* will not complete, and thus the recursive call

4488 to *pthread\_once()* will not return. Use of *longjmp()*, *\_longjmp()*, or *siglongjmp()* within an *init\_routine* to  
4489 jump to a point outside of *init\_routine* prevents *init\_routine* from returning.

4490 *Rationale*: Austin Group Defect Report(s) applied: 863. See <http://austingroupbugs.net/view.php?id=863>.

4491 **Change Number: XSH/TC2/D4/0285** [874]

4492 On Page: 1669 Line: 53628 Section: *pthread\_once()*  
4493 (2013 edition Page: 1684 Line: 54506)

4494 In the RATIONALE section, change from:

4495 `extern int initialize_random();`

4496 to:

4497 `extern void initialize_random(void);`

4498 *Rationale*: Austin Group Defect Report(s) applied: 874. See <http://austingroupbugs.net/view.php?id=874>.

4499 **Change Number: XSH/TC2/D4/0286** [874]

4500 On Page: 1670 Line: 53640 Section: *pthread\_once()*  
4501 (2013 edition Page: 1685 Line: 54518)

4502 In the RATIONALE section, change from:

4503 For dynamic library initialization in a multi-threaded process, a simple initialization flag is not sufficient;  
4504 the flag needs to be protected against modification by multiple threads simultaneously calling into the  
4505 library. Protecting the flag requires the use of a mutex; however, mutexes have to be initialized before they  
4506 are used. Ensuring that the mutex is only initialized once requires a recursive solution to this problem.

4507 The use of *pthread\_once()* not only supplies an implementation-guaranteed means of dynamic  
4508 initialization, it provides an aid to the reliable construction of multi-threaded and realtime systems. The  
4509 preceding example then becomes:

4510 to:

4511 For dynamic library initialization in a multi-threaded process, if an initialization flag is used the flag needs  
4512 to be protected against modification by multiple threads simultaneously calling into the library. This can be  
4513 done by using a mutex (initialized by assigning PTHREAD\_MUTEX\_INITIALIZER). However, the better  
4514 solution is to use *pthread\_once()*, which is designed for exactly this purpose, as follows:

4515 On Page: 1670 Line: 53650 Section: *pthread\_once()*  
4516 (2013 edition Page: 1685 Line: 54528)

4517 In the RATIONALE section, change from:

4518 `extern int initialize_random();`

4519 to:

4520 `extern void initialize_random(void);`

4521 *Rationale:* Austin Group Defect Report(s) applied: 874. See <http://austingroupbugs.net/view.php?id=874>.

4522 **Change Number: XSH/TC2/D4/0287** [747]

4523 On Page: 1670 Line: 53656 Section: `pthread_once()`

4524 In the RATIONALE section, remove the text:

4525 Note that a `pthread_once_t` cannot be an array because some compilers do not accept the construct  
 4526 `&<array_name>`.

4527 *Rationale:* Austin Group Defect Report(s) applied: 747. See <http://austingroupbugs.net/view.php?id=747>.

4528 **Change Number: XSH/TC2/D4/0288** [972]

4529 On Page: 1671 Line: 53698-53701 Section: `pthread_rwlock_destroy()`  
 4530 (2013 edition Page: 1686 Line: 54576-54579)

4531 In the DESCRIPTION section, change from:

4532 Only the object referenced by `rwlock` may be used for performing synchronization. The result of referring  
 4533 to copies of that object in calls to `pthread_rwlock_destroy()`, `pthread_rwlock_rdlock()`,  
 4534 `pthread_rwlock_timedrdlock()`, `pthread_rwlock_timedwrlock()`, `pthread_rwlock_tryrdlock()`,  
 4535 `pthread_rwlock_trywrlock()`, `pthread_rwlock_unlock()`, or `pthread_rwlock_wrlock()` is undefined.

4536 to:

4537 See [xref to section 2.9.9] for further requirements.

4538 *Rationale:* Austin Group Defect Report(s) applied: 972. See <http://austingroupbugs.net/view.php?id=972>.

4539 **Change Number: XSH/TC2/D4/0289** [758]

4540 On Page: 1672 Line: 53728 Section: `pthread_rwlock_destroy()`  
 4541 (2013 edition Page: 1687 Line: 54606)

4542 In the RATIONALE section, change from:

4543 `pthread_rwlock_init()`

4544 to:

4545 `pthread_rwlock_init()`

4546 *Rationale:* Austin Group Defect Report(s) applied: 758. See <http://austingroupbugs.net/view.php?id=758>.

4547

4548 **Change Number: XSH/TC2/D4/0290** [720]

4549 On Page: 1682 Line: 54002 Section: pthread\_rwlock\_trywrlock()

4550 In the DESCRIPTION section, change from:

4551 The calling thread acquires the write lock if no other thread (reader or writer) holds the read-write lock  
 4552 *rwlock*. Otherwise, the thread shall block until it can acquire the lock. The calling thread may deadlock if at  
 4553 the time the call is made it holds the read-write lock (whether a read or write lock).

4554 to:

4555 The calling thread shall acquire the write lock if no thread (reader or writer) holds the read-write lock  
 4556 *rwlock*. Otherwise, if another thread holds the read-write lock *rwlock*, the calling thread shall block until it  
 4557 can acquire the lock. If a deadlock condition occurs or the calling thread already owns the read-write lock  
 4558 for writing or reading, the call shall either deadlock or return EDEADLK.

4559 *Rationale*: Austin Group Defect Report(s) applied: 720. See <http://austingroupbugs.net/view.php?id=720>.  
 4560 It is clear from the description of pthread\_rwlock\_[try]rdlock() and pthread\_rwlock\_unlock() that read  
 4561 locks are recursive. It is clear from the description of pthread\_rwlock\_trywrlock() and  
 4562 pthread\_rwlock\_unlock() that write locks are not recursive. The intention is that pthread\_rwlock\_wrlock()  
 4563 either deadlocks or returns EDEADLK if the caller already holds a write lock, but this is written as two  
 4564 separate "may" clauses (one in the DESCRIPTION and one in the ERRORS section), which creates a  
 4565 loophole that allows pthread\_rwlock\_wrlock() to succeed (as a no-op). However, this behavior would be of  
 4566 no use to applications. In particular, if an application tried to make matching pthread\_rwlock\_unlock() calls  
 4567 for two successful pthread\_rwlock\_wrlock() calls, the first unlock call would set the state to unlocked and  
 4568 the second unlock call would result in undefined behavior.

4569 **Change Number: XSH/TC2/D4/0291** [722]

4570 On Page: 1682 Line: 54006 Section: pthread\_rwlock\_trywrlock()  
 4571 (2013 edition Page: 1697 Line: 54885)

4572 In the DESCRIPTION section, remove the line:

4573 Implementations may favor writers over readers to avoid writer starvation.

4574 *Rationale*: Austin Group Defect Report(s) applied: 722. See <http://austingroupbugs.net/view.php?id=722>.  
 4575 Changes were made to the read locks in Issue 6, that should also be represented in the write lock  
 4576 specification regarding read/write precedence.

4577 **Change Number: XSH/TC2/D4/0292** [972]

4578 On Page: 1689 Line: 54196-54199 Section: pthread\_rwlockattr\_getpshared()  
 4579 (2013 edition Page: 1704 Line: 55075-55078)

4580 In the DESCRIPTION section, change from:

4581 If the *process-shared* attribute is PTHREAD\_PROCESS\_PRIVATE, the read-write lock shall only be  
 4582 operated upon by threads created within the same process as the thread that initialized the read-write lock;  
 4583 if threads of differing processes attempt to operate on such a read-write lock, the behavior is undefined.

4584 to:

4585 See [xref to section 2.9.9] for further requirements.

4586 *Rationale:* Austin Group Defect Report(s) applied: 972. See <http://austingroupbugs.net/view.php?id=972>.

4587 **Change Number: XSH/TC2/D4/0293 [757]**

4588 On Page: 1690 Line: 54240 Section: *pthread\_rwlockattr\_getpshared()*  
 4589 (2013 edition Page: 1705 Line: 55119)

4590 In the CHANGE HISTORY section for Issue 7, change from:

4591 The *pthread\_rwlockattr\_getpshared()* and *pthread\_rwlockattr\_setpshared()* functions are moved from the  
 4592 Threads option.

4593 to:

4594 The *pthread\_rwlockattr\_getpshared()* and *pthread\_rwlockattr\_setpshared()* functions are marked only as  
 4595 part of the Thread Process-Shared Synchronization option as the Threads option is now part of the Base.

4596 *Rationale:* Austin Group Defect Report(s) applied: 757. See <http://austingroupbugs.net/view.php?id=757>.

4597 **Change Number: XSH/TC2/D4/0294 [622]**

4598 On Page: 1694 Line: 54327 Section: *pthread\_setcancelstate()*

4599 In the APPLICATION USAGE section, change from:

4600 None.

4601 to:

4602 In order to write a signal handler for an asynchronous signal which can run safely in a cancellable thread,  
 4603 *pthread\_setcancelstate()* must be used to disable cancellation for the duration of any calls that the signal  
 4604 handler makes which are cancellation points. However, the standard does not permit strictly conforming  
 4605 applications to call *pthread\_setcancelstate()* from a signal handler since it is not currently required to be  
 4606 async-signal-safe. On implementations where *pthread\_setcancelstate()* is not async-signal-safe, alternatives  
 4607 are to ensure either that the corresponding signals are blocked during execution of functions that are not  
 4608 async-cancel-safe or that cancellation is disabled during times when those signals could be delivered.  
 4609 Implementations are strongly encouraged to make *pthread\_setcancelstate()* async-signal-safe.

4610 *Rationale:* Austin Group Defect Report(s) applied: 622. See <http://austingroupbugs.net/view.php?id=622>.

4611 **Change Number: XSH/TC2/D4/0295 [615]**

4612 On Page: 1695 Line: 54349 Section: *pthread\_setcancelstate()*

4613 In the FUTURE DIRECTIONS section, change from:

4614 None.

4615 to:

4616 The *pthread\_setcancelstate()* function may be added to the table of async-signal-safe functions in section  
4617 2.4.3 on page 489.

4618 *Rationale*: Austin Group Defect Report(s) applied: 615. See <http://austingroupbugs.net/view.php?id=615>.  
4619 In order to write a signal handler for an asynchronous signal which can run safely in a cancellable thread,  
4620 *pthread\_setcancelstate()* must be used to disable cancellation for the duration of any calls that the signal  
4621 handler makes which are cancellation points. However, the standard does not currently permit strictly  
4622 conforming applications to do this since *pthread\_setcancelstate()* is not required to be async-signal-safe.

4623 **Change Number: XSH/TC2/D4/0296 [757]**

4624 On Page: 1699 Line: 54427 Section: *pthread\_setschedprio()*  
4625 (2013 edition Page: 1714 Line: 55305)

4626 In the CHANGE HISTORY section for Issue 7, change from:

4627 The *pthread\_setschedprio()* function is moved from the Threads option.

4628 to:

4629 The *pthread\_setschedprio()* function is marked only as part of the Thread Execution Scheduling option as  
4630 the Threads option is now part of the Base.

4631 *Rationale*: Austin Group Defect Report(s) applied: 757. See <http://austingroupbugs.net/view.php?id=757>.

4632 **Change Number: XSH/TC2/D4/0297 [972]**

4633 On Page: 1705 Line: 54589-54593 Section: *pthread\_spin\_destroy()*  
4634 (2013 edition Page: 1720 Line: 55469-55473)

4635 In the DESCRIPTION section, change from:

4636 [TSH]If the Thread Process-Shared Synchronization option is supported and the value of *pshared* is  
4637 PTHREAD\_PROCESS\_PRIVATE,[/TSH] or if the option is not supported, the spin lock shall only be  
4638 operated upon by threads created within the same process as the thread that initialized the spin lock. If  
4639 threads of differing processes attempt to operate on such a spin lock, the behavior is undefined.

4640 to:

4641 See [xref to section 2.9.9] for further requirements.

4642 *Rationale*: Austin Group Defect Report(s) applied: 972. See <http://austingroupbugs.net/view.php?id=972>.

4643 **Change Number: XSH/TC2/D4/0298 [503]**

4644 On Page: 1712 Line: 54750 Section: *ptsname()*

4645 In the RETURN VALUE section, change from:

4646 Upon failure *ptsname()* shall return a null pointer.

4647 to:

4648 Upon failure *ptsname()* shall return a null pointer and may set *errno*.

4649 *Rationale*: Austin Group Defect Report(s) applied: 503. See <http://austingroupbugs.net/view.php?id=503>.

4650 **Change Number: XSH/TC2/D4/0299** [656]

4651 On Page: 1712 Line: 54752 Section: *ptsname()*

4652 In the RETURN VALUE section, add a new paragraph at the end of the section:

4653 The application shall not modify the string returned. The returned pointer might be invalidated or the string content might be overwritten by a subsequent call to *ptsname()*. The returned pointer and the string content might also be invalidated if the calling thread is terminated.

4654

4655

4656 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.

4657 This item is a layered change on XSH/TC1/D5/0468 [75].

4658 The change is to add the following text to the end of the paragraph at 2013 edition P1727, L55633-55634:

4659 The returned pointer and the string content might also be invalidated if the calling thread is terminated.

4660 Note that the other change in XSH/TC1/D5/0468 [75] to the APPLICATION USAGE section is not

4661 impacted.

4662 **Change Number: XSH/TC2/D4/0300** [503]

4663 On Page: 1712 Line: 54754 Section: *ptsname()*

4664 In the ERRORS section, change from:

4665 No errors are defined.

4666 to:

4667 The *ptsname()* function may fail if:

4668 [EBADF]

4669 The *fd* argument is not a valid file descriptor.

4670 [ENOTTY]

4671 The file associated with the *fd* argument is not a master pseudo-terminal device.

4672 *Rationale*: Austin Group Defect Report(s) applied: 503. See <http://austingroupbugs.net/view.php?id=503>.

4673

4674 **Change Number: XSH/TC2/D4/0301** [743]

4675 On Page: 1734 Line: 55339 Section: rand()  
4676 (2013 edition Page: 1749 Line: 56250)

4677 In the APPLICATION USAGE section, change from:

4678 The *drand48()* function provides a much more elaborate random number generator.

4679 to:

4680 The *drand48()* and *random()* functions provide much more elaborate pseudo-random number generators.

4681 On Page: 1734 Line: 55342-55343 Section: rand()  
4682 (2013 edition Page: 1749 Line: 56253-56254)

4683 In the APPLICATION USAGE section, change from:

4684 Therefore this function should be avoided whenever non-trivial requirements (including safety) have to be  
4685 fulfilled.

4686 to a new paragraph:

4687 These functions should be avoided whenever non-trivial requirements (including safety) have to be  
4688 fulfilled.

4689 On Page: 1735 Line: 55362 Section: rand()  
4690 (2013 edition Page: 1750 Line: 56273)

4691 In the SEE ALSO section, add *initstate()*.

4692 *Rationale:* Austin Group Defect Report(s) applied: 743. See <http://austingroupbugs.net/view.php?id=743>.

4693 **Change Number: XSH/TC2/D4/0302** [710]

4694 On Page: 1739 Line: 55477 Section: read()

4695 In the DESCRIPTION section, change from:

4696 The *pread()* function shall be equivalent to *read()*, except that it shall read from a given position in the file  
4697 without changing the file pointer.

4698 to:

4699 The *pread()* function shall be equivalent to *read()*, except that it shall read from a given position in the file  
4700 without changing the file offset.

4701 *Rationale:* Austin Group Defect Report(s) applied: 710. See <http://austingroupbugs.net/view.php?id=710>.

4702

4703 **Change Number: XSH/TC2/D4/0303** [676,710]

4704 On Page: 1739 Line: 55506 Section: read()  
 4705 (2013 edition Page: 1754 Line: 56422)

4706 In the ERRORS section, before "The *read()* function shall fail if" insert:

4707 The *pread()* function shall fail if:

4708     [EINVAL]  
 4709         The file is a regular file or block special file, and the offset argument is negative. The file  
 4710         offset shall remain unchanged.

4711     [ESPIPE]  
 4712         The file is incapable of seeking.

4713 On Page: 1740 Line: 55520 Section: read()

4714 In the ERRORS section, delete lines 55520-55525 (the *pread()* "shall fail" errors).

4715 *Rationale:* Austin Group Defect Report(s) applied: 676,710. See  
<http://austingroupbugs.net/view.php?id=676>.  
 4717 See <http://austingroupbugs.net/view.php?id=710>.  
 4718 This is a layered change on XSH/TC1/D5/0484 [218] and XSH/TC1/D5/0482 [218].

4719 **Change Number: XSH/TC2/D4/0304** [656]

4720 On Page: 1744 Line: 55685 Section: readdir()

4721 In the DESCRIPTION section, change from:

4722 The pointer returned by *readdir()* points to data which may be overwritten by another call to *readdir()* on  
 4723 the same directory stream. This data is not overwritten by another call to *readdir()* on a different directory  
 4724 stream.

4725 to:

4726 The application shall not modify the structure to which the return value of *readdir()* points, nor any storage  
 4727 areas pointed to by pointers within the structure. The returned pointer, and pointers within the structure,  
 4728 might be invalidated or the structure or the storage areas might be overwritten by a subsequent call to  
 4729 *readdir()* on the same directory stream. They shall not be affected by a call to *readdir()* on a different  
 4730 directory stream. The returned pointer, and pointers within the structure, might also be invalidated if the  
 4731 calling thread is terminated.

4732 *Rationale:* Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 4733 This item is a layered change on XSH/TC1/D5/0486 [75].

4734 The change is to add the following text to the end of the paragraph at 2013 edition P1759, L56601-56605:

4735 The returned pointer, and pointers within the structure, might also be invalidated if the calling thread is  
 4736 terminated.

4737 **Change Number: XSH/TC2/D4/0305** [591]

4738 On Page: 1749 Line: 55846 Section: readlink()

4739 Before the readlinkat SYNOPSIS line, insert a line with OH shading:

4740 `#include <fcntl.h>`

4741 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

4742 **Change Number: XSH/TC2/D4/0306** [817]

4743 On Page: 1749 Line: 55858 Section: readlink()

4744 (2013 edition Page: 1764 Line: 56779)

4745 In the DESCRIPTION section, change from:

4746 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
4747 are permitted using the current permissions of the directory underlying the file descriptor. If the file  
4748 descriptor was opened with O\_SEARCH, the function shall not perform the check.

4749 to:

4750 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
4751 function shall check whether directory searches are permitted using the current permissions of the directory  
4752 underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

4753 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

4754 **Change Number: XSH/TC2/D4/0307** [817]

4755 On Page: 1750 Line: 55885 Section: readlink()

4756 (2013 edition Page: 1765 Line: 56805)

4757 In the ERRORS section, for the [EACCES] error, change from:

4758 *fd* was not opened with O\_SEARCH and ...

4759 to:

4760 The access mode of the open file description associated with *fd* is not O\_SEARCH and ...

4761 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

4762 **Change Number: XSH/TC2/D4/0308** [591]

4763 On Page: 1751 Line: 55930 Section: readlink()

4764 Add a reference to <fcntl.h> to the SEE ALSO list.

4765 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

4766 **Change Number: XSH/TC2/D4/0309** [526]

4767 On Page: 1754 Line: 56030 Section: realloc()

4768 In the DESCRIPTION section, change from:

4769 The *realloc()* function shall change the size of the memory object pointed to by *ptr* to the size specified by *size*. The contents of the object shall remain unchanged up to the lesser of the new and old sizes. If the new size of the memory object would require movement of the object, the space for the previous instantiation of the object is freed. If the new size is larger, the contents of the newly allocated portion of the object are unspecified. If *size* is 0 and *ptr* is not a null pointer, the object pointed to is freed.

4774 to:

4775 The *realloc()* function shall deallocate the old object pointed to by *ptr* and return a pointer to a new object  
 4776 that has the size specified by *size*. The contents of the new object shall be the same as that of the old object  
 4777 prior to deallocation, up to the lesser of the new and old sizes. Any bytes in the new object beyond the size  
 4778 of the old object have indeterminate values. If the size of the space requested is zero, the behavior shall be  
 4779 implementation-defined: either a null pointer is returned, or the behavior shall be as if the size were some  
 4780 nonzero value, except that the behavior is undefined if the returned pointer is used to access an object.

4781 *Rationale:* Austin Group Defect Report(s) applied: 526. See <http://austingroupbugs.net/view.php?id=526>.

4782 This change is layered on XSH/TC1/D5/0495 applied in TC1, the difference being that bug report 526  
 4783 changes from:

4784 except that the returned pointer shall not be used to access an object.

4785 to:

4786 except that the behavior is undefined if the returned pointer is used to access an object.

4788 **Change Number: XSH/TC2/D4/0310** [526,688]

4789 On Page: 1754 Line: 56046 Section: realloc()

4790 In the RETURN VALUE section, change from:

4791 Upon successful completion with a size not equal to 0, *realloc()* shall return a pointer to the (possibly  
 4792 moved) allocated space. If *size* is 0, either a null pointer or a unique pointer that can be successfully passed  
 4793 to *free()* shall be returned. If there is not enough available memory, *realloc()* shall return a null pointer  
 4794 [CX]and set *errno* to [ENOMEM].[/CX]

4795 to:

4796 Upon successful completion, *realloc()* shall return a pointer to the (possibly moved) allocated space. If *size*  
 4797 is 0, either:

- 4798 • A null pointer shall be returned [CX]and, if *ptr* is not a null pointer, *errno* shall be set to an  
 4799 implementation defined value[/CX].
- 4800 • A pointer to the allocated space shall be returned, and the memory object pointed to by *ptr* shall be  
 4801 freed. The application shall ensure that the pointer is not used to access an object.

4802 If there is not enough available memory, *realloc()* shall return a null pointer [CX]and set *errno* to  
 4803 [ENOMEM][/CX]. If *realloc()* returns a null pointer [CX]and *errno* has been set to a ENOMEM[/CX], the

4804 memory referenced by *ptr* shall not be changed.

4805 *Rationale*: Austin Group Defect Report(s) applied: 526,688. See <http://austingroupbugs.net/view.php?id=526>.

4806 See <http://austingroupbugs.net/view.php?id=688>.

4807 This change is layered on XSH/TC1/D5/0496 [400] applied in TC1, the difference being that bug report 526 changes from:

4810 A unique pointer that can be successfully passed to *free()* shall be returned to:

4812 A pointer to the allocated space shall be returned

4813 This is also a layered change on XSH/TC1/D5/0496 [400] as a result of bug report 688. The difference being that the wording is changed from:

4815 A null pointer shall be returned [CX] and *errno* set to an implementation-defined value[/CX].

4816 to:

4817 A null pointer shall be returned [CX] and, if *ptr* is not a null pointer, *errno* shall set to an implementation-defined value[/CX].

4819 **Change Number: XSH/TC2/D4/0311** [873]

4820 On Page: 1781 Line: 56918 Section: *rename()*

4821 (2013 edition Page: 1797 Line: 57881)

4822 In the NAME section, delete:

4823 relative to directory file descriptor

4824 *Rationale*: Austin Group Defect Report(s) applied: 873. See <http://austingroupbugs.net/view.php?id=873>.

4825 **Change Number: XSH/TC2/D4/0312** [591]

4826 On Page: 1781 Line: 56922 Section: *rename()*

4827 Before the *renameat* SYNOPSIS line, insert a line with OH CX shading:

4828 `#include <fcntl.h>`

4829 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

4830 **Change Number: XSH/TC2/D4/0313** [716]

4831 On Page: 1781 Line: 56939 Section: *rename()*

4832 In the DESCRIPTION section, change from:

4833 In this case, a link named *new* shall remain visible to other processes throughout the renaming operation

4834 to:

4835 In this case, a link named *new* shall remain visible to other threads throughout the renaming operation

4836 *Rationale*: Austin Group Defect Report(s) applied: 716. See <http://austingroupbugs.net/view.php?id=716>.

4837

4838 **Change Number: XSH/TC2/D4/0314** [817]

4839 On Page: 1782 Line: 56969 Section: rename()  
 4840 (2013 edition Page: 1798 Line: 57932)

4841 In the DESCRIPTION section, change from:

4842 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
 4843 are permitted using the current permissions of the directory underlying the file descriptor. If the file  
 4844 descriptor was opened with O\_SEARCH, the function shall not perform the check.

4845 to:

4846 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
 4847 function shall check whether directory searches are permitted using the current permissions of the directory  
 4848 underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

4849 *Rationale:* Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

4850 **Change Number: XSH/TC2/D4/0315** [817]

4851 On Page: 1783 Line: 57023 Section: rename()  
 4852 (2013 edition Page: 1799 Line: 57990)

4853 In the ERRORS section, for the [EACCES] error, change from:

4854 *oldfd* or *newfd* was not opened with O\_SEARCH and ...

4855 to:

4856 The access mode of the open file description associated with *oldfd* or *newfd* is not O\_SEARCH and ...

4857 *Rationale:* Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

4858 **Change Number: XSH/TC2/D4/0316** [591]

4859 On Page: 1784 Line: 57083 Section: rename()

4860 Add a reference to <fcntl.h> to the SEE ALSO list.

4861 *Rationale:* Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

4862 **Change Number: XSH/TC2/D4/0317** [870]

4863 On Page: 1812 Line: 57894 Section: sem\_close()  
 4864 (2013 edition Page: 1827 Line: 58865)

4865 In the DESCRIPTION section, change from:

4866 The effect of subsequent use of the semaphore indicated by *sem* by this process is undefined.

4867 to:

4868 The effect of subsequent use of the semaphore indicated by *sem* by this process is undefined. If any threads  
4869 in the calling process are currently blocked on the semaphore, the behavior is undefined.

4870 *Rationale*: Austin Group Defect Report(s) applied: 870. See <http://austingroupbugs.net/view.php?id=870>.  
4871 There are current implementations which return *mmap*()ed memory and therefore a *sem\_close*() in this  
4872 situation on such implementations will result in undefined behavior.

4873 **Change Number: XSH/TC2/D4/0318** [972]

4874 On Page: 1818 Line: 58029-58031 Section: *sem\_init*()  
4875 (2013 edition Page: 1832 Line: 59000-59002)

4876 In the DESCRIPTION section, delete:

4877 Only *sem* itself may be used for performing synchronization. The result of referring to copies of *sem* in  
4878 calls to *sem\_wait*(), *sem\_timedwait*(), *sem\_trywait*(), *sem\_post*(), and *sem\_destroy*() is undefined.

4879 On Page: 1818 Line: 58034-58035 Section: *sem\_init*()  
4880 (2013 edition Page: 1832 Line: 59005-59006)

4881 In the DESCRIPTION section, change from:

4882 The use of the semaphore by threads other than those created in the same process is undefined.

4883 to (as a new paragraph):

4884 See [xref to section 2.9.9] for further requirements.

4885 *Rationale*: Austin Group Defect Report(s) applied: 972. See <http://austingroupbugs.net/view.php?id=972>.

4886 **Change Number: XSH/TC2/D4/0319** [532]

4887 On Page: 1833 Line: 58506 Section: *semctl*()

4888 In the DESCRIPTION section add the following new paragraph after the union *semun* definition:

4889 Each operation shall be performed atomically.

4890 *Rationale*: Austin Group Defect Report(s) applied: 532. See <http://austingroupbugs.net/view.php?id=532>.

4891 **Change Number: XSH/TC2/D4/0320** [899]

4892 On Page: 1867 Line: 59571 Section: *setkey*()  
4893 (2013 edition Page: 1882 Line: 60535)

4894 In the FUTURE DIRECTIONS section, change from:

4895 None.

4896 to:

4897 A future version of the standard may mark this interface as obsolete or remove it altogether.

4898 *Rationale:* Austin Group Defect Report(s) applied: 899. See <http://austingroupbugs.net/view.php?id=899>.

4899 **Change Number: XSH/TC2/D4/0321** [826]

4900 On Page: 1869 Line: 59629 Section: setlocale()  
 4901 (2013 edition Page: 1884 Line: 60595)

4902 In the DESCRIPTION section, add a new paragraph to the end of the section:

4903 [CX]The *setlocale()* function need not be thread-safe.[/CX]

4904 *Rationale:* Austin Group Defect Report(s) applied: 826. See <http://austingroupbugs.net/view.php?id=826>.

4905 **Change Number: XSH/TC2/D4/0322** [826]

4906 On Page: 1869 Line: 59637 Section: setlocale()  
 4907 (2013 edition Page: 1884 Line: 60604)

4908 In the RETURN VALUE section, change from:

4909 The application shall not modify the string returned which may be overwritten by a subsequent call to  
 4910 *setlocale()*.

4911 to:

4912 The application shall not modify the string returned. [CX]The returned string pointer might be invalidated  
 4913 or[/CX] the string content might be overwritten by a subsequent call to *setlocale()*. [CX]The returned  
 4914 pointer might also be invalidated if the calling thread is terminated.[/CX]

4915 *Rationale:* Austin Group Defect Report(s) applied: 826. See <http://austingroupbugs.net/view.php?id=826>.  
 4916 This is a layered change, building upon XSH/TC1/D5/0567 [288], the difference being the addition of:  
 4917 [CX]The returned pointer might also be invalidated if the calling thread is terminated.[/CX]

4918 **Change Number: XSH/TC2/D4/0323** [596]

4919 On Page: 1871 Line: 59720 Section: setlocale()

4920 In the SEE ALSO section, remove the *setlocale()* entry.

4921 *Rationale:* Austin Group Defect Report(s) applied: 596. See <http://austingroupbugs.net/view.php?id=596>.

4922 **Change Number: XSH/TC2/D4/0324** [835]

4923 On Page: 1898 Line: 60324 Section: *shm\_open()*  
 4924 (2013 edition Page: 1913 Line: 61308)

4925 Change from:

4926 The file descriptor is used by other functions to refer to that shared memory object.

4927 to:

4928 The file descriptor shall be allocated as described in [xref to new section 2.14] and can be used by other  
4929 functions to refer to that shared memory object.

4930 *Rationale:* Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

4931 **Change Number: XSH/TC2/D4/0325** [835]

4932 On Page: 1898 Line: 60335 Section: `shm_open()`  
4933 (2013 edition Page: 1913 Line: 61319)

4934 Change from:

4935 ... return a file descriptor for the shared memory object that is the lowest numbered file descriptor not  
4936 currently open for that process.

4937 to:

4938 ... return a file descriptor for the shared memory object.

4939 *Rationale:* Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

4940 **Change Number: XSH/TC2/D4/0326** [835]

4941 On Page: 1899 Line: 60375 Section: `shm_open()`  
4942 (2013 edition Page: 1914 Line: 61358)

4943 Change from:

4944 ... return a non-negative integer representing the lowest numbered unused file descriptor.

4945 to:

4946 ... return a non-negative integer representing the file descriptor.

4947 *Rationale:* Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

4948 **Change Number: XSH/TC2/D4/0327** [522]

4949 On Page: 1905 Line: 60591 Section: `shmat()`

4950 In the RETURN VALUE section, change from:

4951 `shmat()` shall return -1

4952 to:

4953 *shmat()* shall return **(void\*)-1**

4954 *Rationale:* Austin Group Defect Report(s) applied: 522. See <http://austingroupbugs.net/view.php?id=522>.

4955 **Change Number: XSH/TC2/D4/0328** [640]

4956 On Page: 1912 Line: 60779-60780 Section: *shmget()*

4957 In the ERRORS section, change from:

4958 A shared memory identifier and associated shared memory segment shall be created, but the amount of  
 4959 available physical memory is not sufficient to fill the request.

4960 to:

4961 A shared memory identifier and associated shared memory segment are to be created, but the amount of  
 4962 available physical memory is not sufficient to fill the request.

4963 *Rationale:* Austin Group Defect Report(s) applied: 640. See <http://austingroupbugs.net/view.php?id=640>.

4964 **Change Number: XSH/TC2/D4/0329** [690]

4965 On Page: 1917 Line: 60935 Section: *sigaction()*  
 4966 (2013 edition Page: 1932 Line: 61931)

4967 In the DESCRIPTION section change from:

4968 **SA\_NOCLDWAIT:** If set, and sig equals SIGCHLD, child processes of the calling processes shall not be  
 4969 transformed into zombie processes when they terminate. If the calling process subsequently waits for its  
 4970 children, and the process has no unwaited-for children that were transformed into zombie processes, it shall  
 4971 block until all of its children terminate, and *wait()*, *waitid()*, and *waitpid()* shall fail and set *errno* to  
 4972 [ECHILD]. Otherwise, terminating child processes shall be transformed into zombie processes, unless  
 4973 **SIGCHLD** is set to **SIG\_IGN**.

4974 to:

4975 **[XSI]SA\_NOCLDWAIT:** If sig does not equal SIGCHLD, the behavior is unspecified. Otherwise, the  
 4976 behavior of the SA\_NOCLDWAIT flag is as specified under "Consequences of Process Termination" in the  
 4977 description of the *\_Exit()* function on page 549. **[/XSI]**

4978 (Note the addition of XSI shading.)

4979 *Rationale:* Austin Group Defect Report(s) applied: 690. See <http://austingroupbugs.net/view.php?id=690>.

4980 **Change Number: XSH/TC2/D4/0330** [491]

4981 On Page: 1918 Line: 60981 Section: *sigaction()*

4982 In the ERRORS section, delete the [ENOTSUP] error:

4983 [ENOTSUP]

4984 The `SA_SIGINFO` bit flag is set in the `sa_flags` field of the **`sigaction`** structure.

4985 On Page: 1918 Line: 60985 Section: `sigaction()`

4986

4987 In the ERRORS section, change from:

4988

4989 In addition, the `sigaction()` function may fail if the `SA_SIGINFO` flag is set ...

4990

4991 to:

4992

4993 In addition, on systems that do not support the XSI option, the `sigaction()` function may fail if the

4994 `SA_SIGINFO` flag is set ...

4995 *Rationale*: Austin Group Defect Report(s) applied: 491. See <http://austingroupbugs.net/view.php?id=491>.

4996 **Change Number: XSH/TC2/D4/0331** [785]

4997 On Page: 1937 Line: 61615 Section: `signal()`

4998 In the DESCRIPTION section, delete the paragraph:

4999 Use of this function is unspecified in a multi-threaded process.

5000 *Rationale*: Austin Group Defect Report(s) applied: 785. See <http://austingroupbugs.net/view.php?id=785>.

5001 **Change Number: XSH/TC2/D4/0332** [844]

5002 On Page: 1944 Line: 61771 Section: `sigqueue()`

5003 (2013 edition Page: 1959 Line: 62775)

5004 In the SYNOPSIS section, change from:

5005 `const union sigval value`

5006 to:

5007 `union sigval value`

5008 *Rationale*: Austin Group Defect Report(s) applied: 844. See <http://austingroupbugs.net/view.php?id=844>.

5009 **Change Number: XSH/TC2/D4/0333** [815]

5010 On Page: 1951 Line: 61997 Section: `sigtimedwait()`

5011 In the DESCRIPTION section, change from:

5012 The `sigwaitinfo()` function shall be equivalent to the `sigwait()` function if the `info` argument is `NULL`. If the

5013 `info` argument is non-`NULL`, the `sigwaitinfo()` function shall be equivalent to `sigwait()`, except that the

5014 selected signal number ...

5015 to:

5016 The *sigwaitinfo()* function shall be equivalent to the *sigwait()* function, except that the return value and the  
 5017 error reporting method are different (see RETURN VALUE), and that if the *info* argument is non-NULL,  
 5018 the selected signal number ...

5019 *Rationale:* Austin Group Defect Report(s) applied: 815. See <http://austingroupbugs.net/view.php?id=815>.

5020 **Change Number:** XSH/TC2/D4/0334 [625]

5021 On Page: 1963 Line: 62339 Section: *sleep()*

5022 In the DESCRIPTION section, add a new paragraph after the first paragraph:

5023 In single-threaded programs, *sleep()* may make use of SIGALRM. In multi-threaded programs, *sleep()* shall  
 5024 not make use of SIGALRM and the remainder of this DESCRIPTION does not apply.

5025 In the RATIONALE section on page 1963 line 62369 change from:

5026 This volume of POSIX.1-2008 permits either approach.

5027 to:

5028 This volume of POSIX.1-2008 permits either approach in single-threaded programs, but the simple  
 5029 alarm/suspend implementation is not appropriate for multi-threaded programs.

5030 On Page: 1964 Line: 62406 Section: *sleep()*

5031 In the FUTURE DIRECTIONS section, change from:

5032 None.

5033 to:

5034 A future version of this standard may require that *sleep()* does not make use of SIGALRM in all programs,  
 5035 not just multi-threaded programs.

5036 *Rationale:* Austin Group Defect Report(s) applied: 625. See <http://austingroupbugs.net/view.php?id=625>.

5037 **Change Number:** XSH/TC2/D4/0335 [835]

5038 On Page: 1968 Line: 62491 Section: *socket()*  
 5039 (2013 edition Page: 1983 Line: 63518)

5040 After the text:

5041 ... return a file descriptor that can be used in later function calls that operate on sockets.

5042 add:

5043 The file descriptor shall be allocated as described in [xref to new section 2.14].

5044 *Rationale:* Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

5045 **Change Number: XSH/TC2/D4/0336** [835]

5046 On Page: 1970 Line: 62565 Section: socketpair()  
5047 (2013 edition Page: 1985 Line: 63592)

5048 After the text:

5049 The file descriptors used in referencing the created sockets shall be returned in *socket\_vector*[0] and  
5050 *socket\_vector*[1].

5051 add:

5052 The file descriptors shall be allocated as described in [xref to new section 2.14].

5053 *Rationale*: Austin Group Defect Report(s) applied: 835. See <http://austingroupbugs.net/view.php?id=835>.

5054 **Change Number: XSH/TC2/D4/0337** [483,835]

5055 On Page: 1970 Line: 62593 Section: socketpair()  
5056 (2013 edition Page: 1985 Line: 63619)

5057 Change from:

5058 ... otherwise, -1 shall be returned and *errno* set to indicate the error.

5059 to:

5060 ... otherwise, -1 shall be returned and *errno* set to indicate the error, no file descriptors shall be allocated  
5061 and the contents of *socket\_vector* shall be left unmodified.

5062 *Rationale*: Austin Group Defect Report(s) applied: 483,835. See  
5063 <http://austingroupbugs.net/view.php?id=483>.  
5064 See <http://austingroupbugs.net/view.php?id=835>.

5065 **Change Number: XSH/TC2/D4/0338** [738]

5066 On Page: 1997 Line: 63203 Section: strdup()  
5067 (2013 edition Page: 2012 Line: 64235)

5068 In the APPLICATION USAGE section, add a new paragraph at the end of the section:

5069 Implementations are free to *malloc*() a buffer containing either (*size* + 1) bytes or (*strnlen*(*s*, *size*) + 1)  
5070 bytes. Applications should not assume that *strndup*() will allocate (*size* + 1) bytes when *strlen*(*s*) is smaller  
5071 than *size*.

5072 *Rationale*: Austin Group Defect Report(s) applied: 738. See <http://austingroupbugs.net/view.php?id=738>.

5073 **Change Number: XSH/TC2/D4/0339** [656]

5074 On Page: 1999 Line: 63237 Section: strerror()

5075 In the DESCRIPTION section, change from:

5076 The string pointed to shall not be modified by the application. The string may be overwritten by a  
 5077 subsequent call to *strerror()*. [CX]The string may be overwritten by a subsequent call to *strerror\_l()* in the  
 5078 same thread.[/CX]

5079 to:

5080 The application shall not modify the string returned. [CX]The returned string pointer might be invalidated  
 5081 or[/CX] the string content might be overwritten by a subsequent call to *strerror()*, [CX]or by a subsequent  
 5082 call to *strerror\_l()* in the same thread. The returned pointer and the string content might also be invalidated  
 5083 if the calling thread is terminated.[/CX]

5084 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 5085 This item is a layered change on XSH/TC1/D5/0595 [75].

5086 The change is to add the following text to the end of the paragraph at 2013 edition P2014, L64269-64271:

5087 The returned pointer and the string content might also be invalidated if the calling thread is terminated.

5088 **Change Number: XSH/TC2/D4/0340** [584]

5089 On Page: 2007 Line: 63525 Section: *strftime()*

5090 In the DESCRIPTION section, change from:

5091 minus-sign character ('-')

5092 to:

5093 <hyphen-minus> character ('-')

5094 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

5095 **Change Number: XSH/TC2/D4/0341** [796]

5096 On Page: 2012 Line: 63745 Section: *strftime()*  
 5097 (2013 edition Page: 2027 Line: 64791)

5098 In the APPLICATION USAGE section, change from:

5099 In the C locale, the E and O modifiers are ignored and the replacement strings for the following

5100 to:

5101 In the C or POSIX locale, the E and O modifiers are ignored and the replacement strings for the following

5102 *Rationale*: Austin Group Defect Report(s) applied: 796. See <http://austingroupbugs.net/view.php?id=796>.

5103 **Change Number: XSH/TC2/D4/0342** [584]

5104 On Page: 2014 Line: 63807 Section: strftime()

5105 In the RATIONALE section, change from:

5106 leading minus-sign for %F,

5107 to:

5108 leading <hyphen-minus> for %F,

5109 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

5110 **Change Number: XSH/TC2/D4/0343** [584]

5111 On Page: 2014 Line: 63815 Section: strftime()

5112 In the RATIONALE section, change from:

5113 leading minus-sign ('-') when using %Y

5114 to:

5115 leading <hyphen-minus> ('-') when using %Y

5116 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

5117 **Change Number: XSH/TC2/D4/0344** [560]

5118 On Page: 2016 Line: 63886 Section: strlen()

5119 In the DESCRIPTION section, change from:

5120 the terminating NUL character

5121 to:

5122 any terminating NUL character

5123 On Page: 2016 Line: 63891 Section: strlen()

5124 In the RETURN VALUE section, change from:

5125 The *strnlen()* function shall return an integer containing the smaller of either the length of the string pointed  
5126 to by *s* or *maxlen*.

5127 to:

5128 The *strnlen()* function shall return the number of bytes preceding the first null byte in the array to which *s*  
5129 points, if *s* contains a null byte within the first *maxlen* bytes; otherwise, it shall return *maxlen*.

5130 *Rationale*: Austin Group Defect Report(s) applied: 560. See <http://austingroupbugs.net/view.php?id=560>.

5131 **Change Number: XSH/TC2/D4/0345** [920]

5132 On Page: 2026 Line: 64133 Section: strftime()  
 5133 (2013 edition Page: 2041 Line: 65181)

5134 In the DESCRIPTION section, change from:

5135 ... conversion specifier other than C, F, or Y.

5136 to:

5137 ... conversion specifier other than C or Y.

5138 *Rationale*: Austin Group Defect Report(s) applied: 920. See <http://austingroupbugs.net/view.php?id=920>.  
 5139 The standard states that the behavior of field widths is unspecified for conversions other than C, F, or Y,  
 5140 but does not specify an F conversion.

5141 **Change Number: XSH/TC2/D4/0346** [919]

5142 On Page: 2029 Line: 64233 Section: strftime()  
 5143 (2013 edition Page: 2044 Line: 65281)

5144 In the EXAMPLES section, change from:

5145 **Data-Plus-Time**

5146 to:

5147 **Date-Plus-Time**

5148 *Rationale*: Austin Group Defect Report(s) applied: 919. See <http://austingroupbugs.net/view.php?id=919>.

5149 **Change Number: XSH/TC2/D4/0347** [656]

5150 On Page: 2032 Line: 64344 Section: strsignal()

5151 In the DESCRIPTION section, change from:

5152 The string pointed to shall not be modified by the application, but may be overwritten by a subsequent call  
 5153 to *strsignal()* or *setlocale()*.

5154 to:

5155 The application shall not modify the string returned. The returned pointer might be invalidated or the string  
 5156 content might be overwritten by a subsequent call to *strsignal()* or *setlocale()*. The returned pointer might  
 5157 also be invalidated if the calling thread is terminated.

5158 *Rationale*: Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.  
 5159 This item is a layered change on XSH/TC1/D5/0609 [75].

5160 The change is to add the following text to the end of the paragraph at 2013 edition P2047, L65392-65393:

5161 The returned pointer might also be invalidated if the calling thread is terminated.

5162 **Change Number: XSH/TC2/D4/0348** [584]

5163 On Page: 2036 Line: 64482 Section: strtod()

5164 In the DESCRIPTION section, change from:

5165 If the subject sequence begins with a minus-sign,

5166 to:

5167 If the subject sequence begins with a <hyphen-minus>,

5168 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

5169 **Change Number: XSH/TC2/D4/0349** [796]

5170 On Page: 2036 Line: 64495 Section: strtod()

5171 (2013 edition Page: 2051 Line: 65544)

5172 Change from:

5173 In other than the C [CX] or POSIX[/CX] locale[CX]s[/CX], other implementation-defined subject  
5174 sequences may be

5175 to:

5176 In other than the C [CX] or POSIX[/CX] locale, additional locale-specific subject sequence forms may be

5177 *Rationale:* Austin Group Defect Report(s) applied: 796. See <http://austingroupbugs.net/view.php?id=796>.

5178 **Change Number: XSH/TC2/D4/0350** [878]

5179 On Page: 2040 Line: 64631-64633 Section: strtok()

5180 (2013 edition Page: 2056 Line: 65689-65691)

5181 In the SYNOPSIS section, change from:

5182 `char *strtok(char *restrict s1, const char *restrict s2);`  
5183 `char *strtok_r(char *restrict s, const char *restrict sep,`  
5184 `char **restrict lasts);`

5185 to:

5186 `char *strtok(char *restrict s, const char *restrict sep);`  
5187 `char *strtok_r(char *restrict s, const char *restrict sep,`  
5188 `char **restrict state);`

5189 On Page: 2040 Line: 64638-64650 Section: strtok()  
 5190 (2013 edition Page: 2056 Line: 65696-65708)

5191 In the second, third and fourth paragraphs of the DESCRIPTION section, replace all instances of *s1* with *s*  
 5192 and *s2* with *sep*.

5193 On Page: 2040 Line: 64656-64667 Section: strtok()  
 5194 (2013 edition Page: 2056 Line: 65714-65725)

5195 In the DESCRIPTION section, change from:

5196 The *strtok\_r()* function considers the null-terminated string *s* as a sequence of zero or more text tokens  
 5197 separated by spans of one or more characters from the separator string *sep*. The argument *lasts* points to a  
 5198 user-provided pointer which points to stored information necessary for *strtok\_r()* to continue scanning the  
 5199 same string.

5200 In the first call to *strtok\_r()*, *s* points to a null-terminated string, *sep* to a null-terminated string of separator  
 5201 characters, and the value pointed to by *lasts* is ignored. The *strtok\_r()* function shall return a pointer to the  
 5202 first character of the first token, write a null character into *s* immediately following the returned token, and  
 5203 update the pointer to which *lasts* points.

5204 In subsequent calls, *s* is a null pointer and *lasts* shall be unchanged from the previous call so that  
 5205 subsequent calls shall move through the string *s*, returning successive tokens until no tokens remain. The  
 5206 separator string *sep* may be different from call to call. When no token remains in *s*, a null pointer shall be  
 5207 returned.

5208 to:

5209 The *strtok\_r()* function shall be equivalent to *strtok()*, except that *strtok\_r()* shall be thread-safe and the  
 5210 argument *state* points to a user-provided pointer that allows *strtok\_r()* to maintain state between calls which  
 5211 scan the same string. The application shall ensure that the pointer pointed to by *state* is unique for each  
 5212 string (*s*) being processed concurrently by *strtok\_r()* calls. The application need not initialize the pointer  
 5213 pointed to by *state* to any particular value. The implementation shall not update the pointer pointed to by  
 5214 *state* to point (directly or indirectly) to resources, other than within the string *s*, that need to be freed or  
 5215 released by the caller.

5216 On Page: 2041 Line: 64704 Section: strtok()  
 5217 (2013 edition Page: 2057 Line: 65766)

5218 In the APPLICATION USAGE section, insert a new paragraph:

5219 Note that if *sep* is the empty string, *strtok()* and *strtok\_r()* return a pointer to the remainder of the string  
 5220 being tokenized.

5221 *Rationale*: Austin Group Defect Report(s) applied: 878. See <http://austingroupbugs.net/view.php?id=878>.  
 5222 The behavior of *strtok\_r()* with an empty *sep* is unclear.

5223 **Change Number: XSH/TC2/D4/0351** [892]

5224 On Page: 2043 Line: 64737 Section: strtol()  
 5225 (2013 edition Page: 2059 Line: 65800)

5226 In the SYNOPSIS section, change from:

5227 long strtol(const char \*restrict str, char \*\*restrict endptr, int base);  
5228 long long strtoll(const char \*restrict str, char \*\*restrict endptr,  
5229 int base)

5230 to:

5231 long strtol(const char \*restrict nptr, char \*\*restrict endptr, int base);  
5232 long long strtoll(const char \*restrict nptr, char \*\*restrict endptr,  
5233 int base)

5234

IECNORM.COM : Click to view the full PDF of ISO/IEC/IEEE 9945:2009/Cor 2:2017

IEEE Std 1003.1™-2008/Cor 2-2016  
IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)  
Base Specifications, Issue 7—Technical Corrigendum 2

5235 On Page: 2043 Line: 64744 Section: strtol()  
5236 (2013 edition Page: 2059 Line: 65807)

5237 In the DESCRIPTION section, change from:

5238 These functions shall convert the initial portion of the string pointed to by *str* ...

5239 to:

5240 These functions shall convert the initial portion of the string pointed to by *nptr* ...

5241 *Rationale*: Austin Group Defect Report(s) applied: 892. See <http://austingroupbugs.net/view.php?id=892>.

5242 **Change Number: XSH/TC2/D4/0352** [584]

5243 On Page: 2043 Line: 64772 Section: strtol()

5244 In the DESCRIPTION section, change from:

5245 begins with a minus-sign,

5246 to:

5247 begins with a <hyphen-minus>,

5248 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

5249 **Change Number: XSH/TC2/D4/0353** [796]

5250 On Page: 2043 Line: 64775 Section: strtol()  
5251 (2013 edition Page: 2059 Line: 65838)

5252 In the DESCRIPTION section, change from:

5253 In other than the C [CX]or POSIX[/CX] locale[CX]s[/CX], other implementation-defined subject sequences may be

5254 to:

5255 In other than the C [CX]or POSIX[/CX] locale, additional locale-specific subject sequence forms may be

5256 *Rationale*: Austin Group Defect Report(s) applied: 796. See <http://austingroupbugs.net/view.php?id=796>.

5258 **Change Number: XSH/TC2/D4/0354** [892]

5259 On Page: 2044 Line: 64778 Section: strtol()  
5260 (2013 edition Page: 2060 Line: 65841)

5261 In the DESCRIPTION section, change from:

5262 ... the value of *str* is stored ...

5263 to:

5264 ... the value of *nptr* shall be stored ...

5265 *Rationale*: Austin Group Defect Report(s) applied: 892. See <http://austingroupbugs.net/view.php?id=892>.  
5266 This item is a layered change on XSH/TC1/D5/0616 [453].

5267 **Change Number: XSH/TC2/D4/0355** [584]

5268 On Page: 2048 Line: 64876 Section: strtoul()

5269 In the DESCRIPTION section, change from:

5270 begins with a minus-sign,

5271 to:

5272 begins with a <hyphen-minus>,

5273 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

5274 **Change Number: XSH/TC2/D4/0356** [796]

5275 On Page: 2049 Line: 64879 Section: strtoul()  
5276 (2013 edition Page: 2065 Line: 65950)

5277 In the DESCRIPTION section, change from:

5278 In other than the C [CX] or POSIX[/CX] locale[CX]s[/CX], other implementation-defined subject  
5279 sequences may be

5280 to:

5281 In other than the C [CX] or POSIX[/CX] locale, additional locale-specific subject sequence forms may be

5282 *Rationale*: Austin Group Defect Report(s) applied: 796. See <http://austingroupbugs.net/view.php?id=796>.

5283 **Change Number: XSH/TC2/D4/0357** [873]

5284 On Page: 2057 Line: 65044 Section: symlink()  
5285 (2013 edition Page: 2073 Line: 66122)

5286 In the NAME section, delete:

5287 relative to directory file descriptor

5288 *Rationale*: Austin Group Defect Report(s) applied: 873. See <http://austingroupbugs.net/view.php?id=873>.

5289

5290 **Change Number: XSH/TC2/D4/0358** [591]

5291 On Page: 2057 Line: 65048 Section: symlink()

5292 Before the symlinkat SYNOPSIS line, insert a line with OH shading:

5293 `#include <fcntl.h>`

5294 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

5295 **Change Number: XSH/TC2/D4/0359** [641]

5296 On Page: 2057 Line: 65053 Section: symlink()

5297 In the DESCRIPTION section, change from:

5298 The string pointed to by *path1* shall be treated only as a character string and shall not be validated as a  
 5299 pathname.

5300 to:

5301 The string pointed to by *path1* shall be treated only as a string and shall not be validated as a pathname.

5302 *Rationale*: Austin Group Defect Report(s) applied: 641. See <http://austingroupbugs.net/view.php?id=641>.

5303 **Change Number: XSH/TC2/D4/0360** [817]

5304 On Page: 2057 Line: 65073 Section: symlink()  
 5305 (2013 edition Page: 2073 Line: 66152)

5306 In the DESCRIPTION section, change from:

5307 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
 5308 are permitted using the current permissions of the directory underlying the file descriptor. If the file  
 5309 descriptor was opened with O\_SEARCH, the function shall not perform the check.

5310 to:

5311 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
 5312 function shall check whether directory searches are permitted using the current permissions of the directory  
 5313 underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

5314 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

5315 **Change Number: XSH/TC2/D4/0361** [822]

5316 On Page: 2058 Line: 65096 Section: symlink()

5317 In the RETURN VALUE section, add after the [ENOENT] error:

5318 [ENOENT] or [ENOTDIR]

5319 The *path2* argument contains at least one non-<slash> character and ends with one or more  
5320 trailing <slash> characters. If *path2* without the trailing <slash> characters would name an  
5321 existing file, an [ENOENT] error shall not occur.

5322 *Rationale*: Austin Group Defect Report(s) applied: 822. See <http://austingroupbugs.net/view.php?id=822>.

5323 **Change Number: XSH/TC2/D4/0362** [817]

5324 On Page: 2058 Line: 65105 Section: symlink()  
5325 (2013 edition Page: 2074 Line: 66185)

5326 In the ERRORS section, for the [EACCES] error, change from:

5327 *fd* was not opened with O\_SEARCH and ...

5328 to:

5329 The access mode of the open file description associated with *fd* is not O\_SEARCH and ...

5330 *Rationale*: Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

5331 **Change Number: XSH/TC2/D4/0363** [591]

5332 On Page: 2059 Line: 65140 Section: symlink()

5333 Add a reference to <fcntl.h> to the SEE ALSO list.

5334 *Rationale*: Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

5335 **Change Number: XSH/TC2/D4/0364** [752]

5336 On Page: 2063 Line: 65302-65317 Section: sysconf()

5337 In the DESCRIPTION section, move lines 65302-65317 (\_SC\_PAGE\_SIZE to \_SC\_TZNAME\_MAX) to  
5338 before line 65224 (\_SC\_ADVISORY\_INFO).

5339 *Rationale*: Austin Group Defect Report(s) applied: 752. See <http://austingroupbugs.net/view.php?id=752>.  
5340 By moving these lines the table will have a block of variables with no underscore, a block of \_POSIX\_\*

5341 variables, a block of \_POSIX2\_\* variables, and a block of \_XOPEN\_\* variables, with each block sorted by  
5342 the \_SC\_\* names.

5343 **Change Number: XSH/TC2/D4/0365** [627]

5344 On Page: 2072 Line: 65661 Section: system()

5345 In the RATIONALE section, add a new paragraph:

5346 Note also that the above example implementation is not thread-safe. Implementations can provide a thread-  
5347 safe *system()* function, but doing so involves complications such as how to restore the signal dispositions  
5348 for SIGINT and SIGQUIT correctly if there are overlapping calls, and how to deal with cancellation. The  
5349 example above would not restore the signal dispositions and would leak a process ID if cancelled. This

5350 does not matter for a non-thread-safe implementation since cancelling a non-thread-safe function results in  
 5351 undefined behavior (see [xref to 2.9.5.2]). To avoid leaking a process ID, a thread-safe implementation  
 5352 would need to terminate the child process when acting on a cancellation.

5353 *Rationale:* Austin Group Defect Report(s) applied: 627. See <http://austingroupbugs.net/view.php?id=627>.  
 5354 The system() function is not required to be thread-safe, yet the standard requires a cancellation point to  
 5355 occur when executing it.

5356 **Change Number: XSH/TC2/D4/0366 [551]**

5357 On Page: 2097 Line: 66353 Section: tdelete()

5358 In the DESCRIPTION section, after the sentence:

5359 The variable pointed to by *rootp* shall be changed if the deleted node was the root of the tree.

5360 add a new sentence:

5361 If the deleted node was the root of the tree and had no children, the variable pointed to by *rootp* shall be set  
 5362 to a null pointer.

5363 On Page: 2098 Line: 66392-66446 Section: tdelete()

5364 In the EXAMPLES section, replace the example program with:

```
5365 #include <limits.h>
5366 #include <search.h>
5367 #include <stdlib.h>
5368 #include <string.h>
5369 #include <stdio.h>
5370
5371 struct element { /* Pointers to these are stored in the tree. */
5372     int count;
5373     char string[];
5374 };
5375
5376 void *root = NULL; /* This points to the root. */
5377
5378 int main(void)
5379 {
5380     char str[_POSIX2_LINE_MAX+1];
5381     int length = 0;
5382     struct element *elementptr;
5383     void *node;
5384     void print_node(const void *, VISIT, int);
5385     int node_compare(const void *, const void *),
5386         delete_root(const void *, const void *);
5387
5388     while (fgets(str, sizeof(str), stdin)) {
5389         /* Set element. */
5390         length = strlen(str);
5391         if (str[length-1] == '\n')
5392             str[--length] = '\0';
5393         elementptr = malloc(sizeof(struct element) + length + 1);
5394         strcpy(elementptr->string, str);
5395         elementptr->count = 1;
5396         /* Put element into the tree. */
5397     }
5398 }
```

```

5397     node = tsearch((void *)elementptr, &root, node_compare);
5398     if (node == NULL) {
5399         fprintf(stderr,
5400             "tsearch: Not enough space available\n");
5401         exit(EXIT_FAILURE);
5402     }
5403     else if ((*struct element **)node != elementptr) {
5404         /* A node containing the element already exists */
5405         (*struct element **)node->count++;
5406         free(elementptr);
5407     }
5408 }
5409 twalk(root, print_node);
5410
5411 /* Delete all nodes in the tree */
5412 while (root != NULL) {
5413     elementptr = *(struct element **)root;
5414     printf("deleting node: string = %s, count = %d\n",
5415             elementptr->string,
5416             elementptr->count);
5417     tdelete((void *)elementptr, &root, delete_root);
5418     free(elementptr);
5419 }
5420
5421     return 0;
5422 }
5423
5424 /*
5425  * This routine compares two nodes, based on an
5426  * alphabetical ordering of the string field.
5427  */
5428 int
5429 node_compare(const void *node1, const void *node2)
5430 {
5431     return strcmp((const struct element *) node1->string,
5432                   (const struct element *) node2->string);
5433 }
5434
5435 /*
5436  * This comparison routine can be used with tdelete()
5437  * when explicitly deleting a root node, as no comparison
5438  * is necessary.
5439  */
5440 int
5441 delete_root(const void *node1, const void *node2)
5442 {
5443     return 0;
5444 }
5445
5446 /*
5447  * This routine prints out a node, the second time
5448  * twalk encounters it or if it is a leaf.
5449  */
5450 void
5451 print_node(const void *ptr, VISIT order, int level)
5452 {
5453     const struct element *p = *(const struct element **) ptr;
5454
5455     if (order == postorder || order == leaf) {
5456         (void) printf("string = %s, count = %d\n",
5457                     p->string, p->count);
5458     }

```

5459     }

5460   On Page: 2099 Line: 66450 Section: tdelete()

5461   In the APPLICATION USAGE section, change from:

5462   The *tsearch()* function uses

5463   to:

5464   The *twalk()* function uses

5465   *Rationale*: Austin Group Defect Report(s) applied: 551. See <http://austingroupbugs.net/view.php?id=551>.

5466   **Change Number: XSH/TC2/D4/0367** [604]

5467   On Page: 2105 Line: 66587-66604 Section: tgamma()

5468   In the DESCRIPTION section, at line 66587 change from:

5469   These functions shall compute the *gamma()* function of *x*.

5470   to:

5471   These functions shall compute  $\Gamma(x)$  where  $\Gamma(x)$  is defined as  $\int_0^{\infty} e^{-t} t^{x-1} dt$ .

5472   In the RETURN VALUE section, at line 66593 change from:

5473   shall return *Gamma(x)*.

5474   to:

5475   shall return the gamma of *x*.

5476   Before line 66604 insert a new paragraph, shaded MXX:

5477   If *x* is subnormal and  $1/x$  is representable,  $1/x$  should be returned.

5478   *Rationale*: Austin Group Defect Report(s) applied: 604. See <http://austingroupbugs.net/view.php?id=604>.

5479   **Change Number: XSH/TC2/D4/0368** [630]

5480   On Page: 2106 Line: 66639 Section: tgamma()

5481   In the APPLICATION USAGE section, delete:

5482   For IEEE Std 754-1985 **double**, overflow happens when  $0 < x < 1/\text{DBL\_MAX}$ , and  $171.7 < x$ .

5483   *Rationale*: Austin Group Defect Report(s) applied: 630. See <http://austingroupbugs.net/view.php?id=630>.

5484 **Change Number: XSH/TC2/D4/0369** [659]

5485 On Page: 2113 Line: 66876 Section: timer\_delete()

5486 In the DESCRIPTION section, add a new paragraph to the end of the section:

5487 The behavior is undefined if the value specified by the *timerid* argument does not correspond to a timer ID  
5488 returned by *timer\_create()* but not yet deleted by *timer\_delete()*.

5489 On Page: 2113 Line: 66881-66882 Section: timer\_delete()

5490 In the ERRORS section, replace the entire section with:

5491 No errors are defined.

5492 On Page: 2113 Line: 66888 Section: timer\_delete()

5493 In the RATIONALE section, change from:

5494 None.

5495 to:

5496 If an implementation detects that the value specified by the *timerid* argument to *timer\_delete()* does not  
5497 correspond to a timer ID returned by *timer\_create()* but not yet deleted by *timer\_delete()*, it is  
5498 recommended that the function should fail and report an [EINVAL] error.

5499 *Rationale:* Austin Group Defect Report(s) applied: 659. See <http://austingroupbugs.net/view.php?id=659>.

5500 **Change Number: XSH/TC2/D4/0370** [659]

5501 On Page: 2115 Line: 66956 Section: timer\_getoverrun()

5502 In the DESCRIPTION section, add a new paragraph to the end of the section:

5503 The behavior is undefined if the value specified by the *timerid* argument to *timer\_getoverrun()*,  
5504 *timer\_gettime()*, or *timer\_settime()* does not correspond to a timer ID returned by *timer\_create()* but not yet  
5505 deleted by *timer\_delete()*.

5506 On Page: 2115 Line: 66968-66970 Section: timer\_getoverrun()

5507 In the ERRORS section, delete:

5508 These functions may fail if:

5509 [EINVAL] The *timerid* argument does not correspond to an ID returned by *timer\_create()* but not yet  
5510 deleted by *timer\_delete()*.

5511 On Page: 2116 Line: 66997 Section: timer\_getoverrun()

5512 In the RATIONALE section, add a new paragraph to the end of the section:

5513 If an implementation detects that the value specified by the *timerid* argument to *timer\_getoverrun()*,  
5514 *timer\_gettime()*, or *timer\_settime()* does not correspond to a timer ID returned by *timer\_create()* but not yet  
5515 deleted by *timer\_delete()*, it is recommended that the function should fail and report an [EINVAL] error.

5516 *Rationale*: Austin Group Defect Report(s) applied: 659. See <http://austingroupbugs.net/view.php?id=659>.

5517 **Change Number: XSH/TC2/D4/0371** [644]

5518 On Page: 2117 Line: 67056 Section: *times()*

5519 In the ERRORS section, change from:

5520 No errors are defined.

5521 to:

5522 The *times()* function shall fail if:

5523 [EOVERFLOW] The return value would overflow the range of *clock\_t*.

5524 *Rationale*: Austin Group Defect Report(s) applied: 644. See <http://austingroupbugs.net/view.php?id=644>.

5525 **Change Number: XSH/TC2/D4/0372** [678]

5526 On Page: 2121 Line: 67134 Section: *tmpfile()*

5527 In the DESCRIPTION section, change from:

5528 The *tmpfile()* function shall create a temporary file and open a corresponding stream. The file shall be  
5529 automatically deleted when all references to the file are closed. The file is opened as in *fopen()* for update  
5530 (*w+*), except that implementations may restrict the permissions, either by clearing the file mode bits or  
5531 setting them to the value *S\_IRUSR* | *S\_IWUSR*.

5532 to:

5533 The *tmpfile()* function shall create a temporary file and open a corresponding stream. The file shall be  
5534 automatically deleted when all references to the file are closed. The file shall be opened as in *fopen()* for update  
5535 (*wb+*), except that implementations may restrict the permissions, either by clearing the file mode  
5536 bits or setting them to the value *S\_IRUSR* | *S\_IWUSR*.

5537 *Rationale*: Austin Group Defect Report(s) applied: 678. See <http://austingroupbugs.net/view.php?id=678>.

5538 **Change Number: XSH/TC2/D4/0373** [685]

5539 On Page: 2131 Line: 67438 Section: *towlower()*

5540 In the DESCRIPTION section, change from:

5541 a wide-character code corresponding to a valid character in the current locale

5542 to:

5543 a wide-character code corresponding to a valid character in the locale used by the function

5544 *Rationale:* Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

5545 **Change Number: XSH/TC2/D4/0374** [685]

5546 On Page: 2133 Line: 67487 Section: towupper()

5547 In the DESCRIPTION section, change from:

5548 a wide-character code corresponding to a valid character in the current locale

5549 to:

5550 a wide-character code corresponding to a valid character in the locale used by the function

5551 *Rationale:* Austin Group Defect Report(s) applied: 685. See <http://austingroupbugs.net/view.php?id=685>.

5552 **Change Number: XSH/TC2/D4/0375** [489]

5553 On Page: 2136 Line: 67570 Section: truncate()

5554 In the DESCRIPTION, change from:

5555 Upon successful completion, if the file size is changed, *truncate()* shall mark for update the last data modification and last file status change timestamps of the file, and the S\_ISUID and S\_ISGID bits of the file mode may be cleared.

5556 to:

5557 Upon successful completion, *truncate()* shall mark for update the last data modification and last file status change timestamps of the file, and the S\_ISUID and S\_ISGID bits of the file mode may be cleared.

5558 *Rationale:* Austin Group Defect Report(s) applied: 489. See <http://austingroupbugs.net/view.php?id=489>.

5559 **Change Number: XSH/TC2/D4/0376** [656]

5560 On Page: 2140 Line: 67654 Section: ttynname()

5561 In the DESCRIPTION section, change from:

5562 The return value may point to static data whose content is overwritten by each call.

5563 to:

5564 The application shall not modify the string returned. The returned pointer might be invalidated or the string content might be overwritten by a subsequent call to *ttynname()*. The returned pointer and the string content might also be invalidated if the calling thread is terminated.

5565 *Rationale:* Austin Group Defect Report(s) applied: 656. See <http://austingroupbugs.net/view.php?id=656>.

5566 This item is a layered change on XSH/TC1/D5/0065 [75,428].

5572 The change is to add the following text to the end of the paragraph at 2013 edition P2159, L68808-68810:

5573 The returned pointer and the string content might also be invalidated if the calling thread is terminated.

5574 **Change Number: XSH/TC2/D4/0377 [880]**

5575 On Page: 2143 Line: 67732 Section: tzset()  
 5576 (2013 edition Page: 2162 Line: 68888)

5577 In the DESCRIPTION section, add a new paragraph to the end of the section:

5578 If a thread accesses *tzname*, [XSI]*daylight*, or *timezone*[/XSI] directly while another thread is in a call to  
 5579 *tzset()*, or to any function that is required or allowed to set timezone information as if by calling *tzset()*, the  
 5580 behavior is undefined.

5581 On Page: 2143 Line: 67747 Section: tzset()  
 5582 (2013 edition Page: 2162 Line: 68903)

5583 In the APPLICATION USAGE section, change from:

5584 None.

5585 to:

5586 Since the *ctime()*, *localtime()*, *mktime()*, *strftime()* and *strftime\_l()* functions are required to set timezone  
 5587 information as if by calling *tzset()*, there is no need for an explicit *tzset()* call before using these functions.  
 5588 However, portable applications should call *tzset()* explicitly before using *ctime\_r()* or *localtime\_r()* because  
 5589 setting timezone information is optional for those functions.

5590 *Rationale:* Austin Group Defect Report(s) applied: 880. See <http://austingroupbugs.net/view.php?id=880>.

5591 **Change Number: XSH/TC2/D4/0378 [873]**

5592 On Page: 2154 Line: 68001 Section: unlink()  
 5593 (2013 edition Page: 2174 Line: 69164)

5594 In the NAME section, delete:

5595 relative to directory file descriptor

5596 *Rationale:* Austin Group Defect Report(s) applied: 873. See <http://austingroupbugs.net/view.php?id=873>.

5597 **Change Number: XSH/TC2/D4/0379 [591]**

5598 On Page: 2154 Line: 68005 Section: unlink()

5599 Before the *unlinkat* SYNOPSIS line, insert a line with OH shading:

5600 `#include <fcntl.h>`

5601 *Rationale:* Austin Group Defect Report(s) applied: 591. See <http://austingroupbugs.net/view.php?id=591>.

5602 **Change Number: XSH/TC2/D4/0380** [817]

5603 On Page: 2154 Line: 68023 Section: unlink()  
 5604 (2013 edition Page: 2174 Line: 69186)

5605 In the DESCRIPTION section, change from:

5606 If the file descriptor was opened without O\_SEARCH, the function shall check whether directory searches  
 5607 are permitted using the current permissions of the directory underlying the file descriptor. If the file  
 5608 descriptor was opened with O\_SEARCH, the function shall not perform the check.

5609 to:

5610 If the access mode of the open file description associated with the file descriptor is not O\_SEARCH, the  
 5611 function shall check whether directory searches are permitted using the current permissions of the directory  
 5612 underlying the file descriptor. If the access mode is O\_SEARCH, the function shall not perform the check.

5613 *Rationale:* Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

5614 **Change Number: XSH/TC2/D4/0381** [817]

5615 On Page: 2155 Line: 68063 Section: unlink()  
 5616 (2013 edition Page: 2175 Line: 69227)

5617 In the ERRORS section, for the [EACCES] error, change from:

5618 *fd* was not opened with O\_SEARCH and ...

5619 to:

5620 The access mode of the open file description associated with *fd* is not O\_SEARCH and ...

5621 *Rationale:* Austin Group Defect Report(s) applied: 817. See <http://austingroupbugs.net/view.php?id=817>.

5622 **Change Number: XSH/TC2/D4/0382** [582]

5623 On Page: 2162 Line: 68286 Section: uselocale()

5624 Replace the DESCRIPTION, RETURN VALUE, and ERRORS sections with:

## 5625 DESCRIPTION

5626 The *uselocale()* function shall set or query the current locale for the calling thread.

5627 The value for the *newloc* argument shall be one of the following:

- 5628 1. A value returned by the *newlocale()* or *duplocale()* functions
- 5629 2. The special locale object descriptor LC\_GLOBAL\_LOCALE
- 5630 3. **(locale\_t)0**

5631 If the *newloc* argument is **(locale\_t)0**, the current locale shall not be changed; this value can

5632       be used to query the current locale setting. If the *newloc* argument is  
 5633       LC\_GLOBAL\_LOCALE, any thread- local locale for the calling thread shall be uninstalled;  
 5634       the thread shall again use the global locale as the current locale, and changes to the global  
 5635       locale shall affect the thread. Otherwise, the locale represented by *newloc* shall be installed as  
 5636       a thread-local locale to be used as the current locale for the calling thread.

5637       Once the *uselocale()* function has been called to install a thread-local locale, the behavior of  
 5638       every interface using data from the current locale shall be affected for the calling thread. The  
 5639       current locale for other threads shall remain unchanged.

5640       **RETURN VALUE**

5641       Upon successful completion, the *uselocale()* function shall return a handle for the thread-  
 5642       local locale that was in use as the current locale for the calling thread on entry to the function,  
 5643       or LC\_GLOBAL\_LOCALE if no thread-local locale was in use. Otherwise, *uselocale()* shall  
 5644       return (locale\_t)0 and set *errno* to indicate the error.

5645       **ERRORS**

5646       The *uselocale()* function may fail if:

5647               [EINVAL]  
 5648               *newloc* is not a valid locale object and is not (locale\_t)0.

5649       *Rationale*: Austin Group Defect Report(s) applied: 582. See <http://austingroupbugs.net/view.php?id=582>.

5650       **Change Number: XSH/TC2/D4/0383** [873]

5651       On Page: 2166 Line: 68405 Section: utimensat()  
 5652       (2013 edition Page: 2187 Line: 69579)

5653       In the NAME section, delete:

5654       relative to directory file descriptor

5655       *Rationale*: Austin Group Defect Report(s) applied: 873. See <http://austingroupbugs.net/view.php?id=873>.

5656       **Change Number: XSH/TC2/D4/0384** [690]

5657       On Page: 2181 Line: 68666 Section: wait()  
 5658       (2013 edition Page: 2203 Line: 69847)

5659       In the DESCRIPTION section change from:

5660       The *wait()* and *waitpid()* functions shall obtain status information pertaining to one of the caller's child  
 5661       processes. Various options permit status information to be obtained for child processes that have terminated  
 5662       or stopped. If status information is available for two or more child processes, the order in which their status  
 5663       is reported is unspecified.

5664  
 5665       The *wait()* function shall suspend execution of the calling thread until status information for one of the  
 5666       terminated child processes of the calling process is available, or until delivery of a signal whose action is  
 5667       either to execute a signal-catching function or to terminate the process. If more than one thread is

5668 suspended in *wait()* or *waitpid()* awaiting termination of the same process, exactly one thread shall return  
 5669 the process status at the time of the target process termination. If status information is available prior to the  
 5670 call to *wait()*, return shall be immediate.

5671 to:

5672 The *wait()* and *waitpid()* functions shall obtain status information (see XSH Section 2.13) pertaining to one  
 5673 of the caller's child processes. The *wait()* function obtains status information for process termination from  
 5674 any child process. The *waitpid()* function obtains status information for process termination, and optionally  
 5675 process stop and/or continue, from a specified subset of the child processes.

5676  
 5677 The *wait()* function shall cause the calling thread to become blocked until status information generated by  
 5678 child process termination is made available to the thread, or until delivery of a signal whose action is either  
 5679 to execute a signal-catching function or to terminate the process, or an error occurs. If termination status  
 5680 information is available prior to the call to *wait()*, return shall be immediate. If termination status  
 5681 information is available for two or more child processes, the order in which their status is reported is  
 5682 unspecified.

5683  
 5684 As described in XSH Section 2.13, the *wait()* and *waitpid()* functions consume the status information they  
 5685 obtain.

5686  
 5687 The behavior when multiple threads are blocked in *wait()*, *waitid()*, or *waitpid()* is described in XSH  
 5688 Section 2.13.

5689 On Page: 2181 Line: 68700-68703 Section: *wait()*  
 5690 (2013 edition Page: 2203 Line: 69881-69884)

5691 In the DESCRIPTION section, delete:

5692 [XSI]If the calling process has SA\_NOCLDWAIT set or has SIGCHLD set to SIG\_IGN, and the process  
 5693 has no unwaited-for children that were transformed into zombie processes, the calling thread shall block  
 5694 until all of the children of the process containing the calling thread terminate, and *wait()* and *waitpid()* shall  
 5695 fail and set *errno* to [ECHILD].[/XSI]

5696 *Rationale:* Austin Group Defect Report(s) applied: 690. See <http://austingroupbugs.net/view.php?id=690>.

5697 **Change Number:** XSH/TC2/D4/0385 [691]

5698 On Page: 2183 Line: 68748-68751,68755-68758 Section: *wait()*  
 5699 (2013 Edition Page: 2205 Line: 69929-69932,69936-69939)

5700 In the DESCRIPTION section, change the XSI shading on the first and third paragraphs on the page to  
 5701 completely enclose those paragraphs.

5702 Remove the reference to *wait()* in the third paragraph, changing it to:

5703 If the information pointed to by *stat\_loc* was stored by a call to *waitpid()* that did not specify the  
 5704 WUNTRACED flag and specified the WCONTINUED flag, exactly one of the macros  
 5705 WIFEXITED(\**stat\_loc*), WIFSIGNALED(\**stat\_loc*), and WIFCONTINUED(\**stat\_loc*) shall evaluate to a  
 5706 non-zero value.

5707 *Rationale:* Austin Group Defect Report(s) applied: 691. See <http://austingroupbugs.net/view.php?id=691>.

5708 **Change Number: XSH/TC2/D4/0386** [690]

5709 On Page: 2186 Line: 68926 Section: wait()

5710 In the APPLICATION USAGE section, add a new paragraph at the end of the section:

5711 [XSI]As specified under "Consequences of Process Termination" in the description of the *\_Exit()* function  
 5712 on page 549, if the calling process has SA\_NOCLDWAIT set or has SIGCHLD set to SIG\_IGN, then the  
 5713 termination of a child process will not cause status information to become available to a thread blocked in  
 5714 *wait()*, *waitid()*, or *waitpid()*. Thus, a thread blocked in one of the wait functions will remain blocked unless  
 5715 some other condition causes the thread to resume execution (such as an [ECHILD] failure due to no  
 5716 remaining children in the set of waited-for children).[/XSI]

5717 *Rationale:* Austin Group Defect Report(s) applied: 690. See <http://austingroupbugs.net/view.php?id=690>.

5718 **Change Number: XSH/TC2/D4/0387** [690]

5719 On Page: 2190 Line: 69058 Section: waitid()  
 5720 (2013 edition Page: 2212 Line: 70244)

5721 In the DESCRIPTION section, change from:

5722 The *waitid()* function shall suspend the calling thread until one child of the process containing the calling  
 5723 thread changes state. It records the current state of a child in the structure pointed to by *infop*. The fields of  
 5724 the structure pointed to by *infop* are filled in as described for the SIGCHLD signal in <signal.h>. If a child  
 5725 process changed state prior to the call to *waitid()*, *waitid()* shall return immediately. If more than one thread  
 5726 is suspended in *wait()*, *waitid()*, or *waitpid()* waiting for termination of the same process, exactly one thread  
 5727 shall return the process status at the time of the target process termination.

5728 to:

5729 The *waitid()* function shall obtain status information (see XSH Section 2.13) pertaining to termination,  
 5730 stop, and/or continue events in one of the caller's child processes.

5731  
 5732 The *waitid()* function shall cause the calling thread to become blocked until an error occurs or status  
 5733 information becomes available to the calling thread that satisfies all of the following properties ("matching  
 5734 status information"):

- 5735 • The status information is from one of the child processes in the set of child processes specified by  
 5736 the *idtype* and *id* arguments.
- 5737 • The state change in the status information matches one of the state change flags set in the options  
 5738 argument.

5739 If matching status information is available prior to the call to *waitid()*, return shall be immediate. If  
 5740 matching status information is available for two or more child processes, the order in which their status is  
 5741 reported is unspecified.

5742  
 5743 As described in XSH Section 2.13, the *waitid()* function consumes the status information it obtains unless  
 5744 the *WNOWAIT* flag is set in the options argument.

5745  
 5746 The behavior when multiple threads are blocked in *wait()*, *waitid()*, or *waitpid()* is described in XSH  
 5747 Section 2.13.

5748  
 5749 The *waitid()* function shall record the obtained status information in the structure pointed to by *infop*. The

5750 fields of the structure pointed to by `infop` shall be filled in as described under "Pointer to a Function" in  
5751 Section 2.4.3 on page 492.

5752 On Page: 2191 Line: 69101 Section: `waitid()`

5753 In the APPLICATION USAGE section, add a new paragraph at the end of the section:

5754 [XSI]As specified under "Consequences of Process Termination" in the description of the `_Exit()` function  
5755 on page 549, if the calling process has `SA_NOCLDWAIT` set or has `SIGCHLD` set to `SIG_IGN`, then the  
5756 termination of a child process will not cause status information to become available to a thread blocked in  
5757 `wait()`, `waitid()`, or `waitpid()`. Thus, a thread blocked in one of the wait functions will remain blocked unless  
5758 some other condition causes the thread to resume execution (such as an [ECHILD] failure due to no  
5759 remaining children in the set of waited-for children).[/XSI]

5760 *Rationale:* Austin Group Defect Report(s) applied: 690. See <http://austingroupbugs.net/view.php?id=690>.

5761 **Change Number:** XSH/TC2/D4/0388 [73]

5762 On Page: 2207 Line: 69521 Section: `wcsftime()`

5763 In the DESCRIPTION section, change from:

5764 • The argument format is a wide-character string and the conversion specifications are replaced by  
5765 corresponding sequences of wide characters.

5766 to:

5767 • The argument format is a wide-character string and the conversion specifications are replaced by  
5768 corresponding sequences of wide characters. It is unspecified whether an encoding error occurs if  
5769 the format string contains `wchar_t` values that do not correspond to members of the character set  
5770 of the current locale.

5771 *Rationale:* Austin Group Defect Report(s) applied: 73. See <http://austingroupbugs.net/view.php?id=73>.

5772 A clarification has been made in the C11 standard.

5774 **Change Number:** XSH/TC2/D4/0389 [740]

5775 On Page: 2207 Line: 69521 Section: `wcsftime()`  
5776 (2013 edition Page: 2229 Line: 70726)

5777 In the DESCRIPTION section, add a new bullet item before the final bullet item:

5778 [CX]Field widths specify the number of wide characters instead of the number of bytes.[/CX]

5779 *Rationale:* Austin Group Defect Report(s) applied: 740. See <http://austingroupbugs.net/view.php?id=740>.

5780

5781 **Change Number: XSH/TC2/D4/0390** [560]

5782 On Page: 2209 Line: 69562-69569 Section: wcslen()

5783 In the DESCRIPTION section, on lines 69562 and 69565 change from:

5784 string

5785 to:

5786 array

5787 On line 69563 change from:

5788 the terminating null wide-character code

5789 to:

5790 any terminating null wide-character code.

5791 In the RETURN VALUE section, on line 69568 change from:

5792 The *wcsnlen()* function shall return an integer containing the smaller of either the length of the wide-  
5793 character string pointed to by *ws* or *maxlen*.

5794 to:

5795 The *wcsnlen()* function shall return the number of wide characters preceding the first null wide character  
5796 code in the array to which *ws* points, if *ws* contains a null wide character code within the first *maxlen* wide  
5797 characters; otherwise, it shall return *maxlen*.

5798 *Rationale*: Austin Group Defect Report(s) applied: 560. See <http://austingroupbugs.net/view.php?id=560>.

5799 **Change Number: XSH/TC2/D4/0391** [584]

5800 On Page: 2224 Line: 69979 Section: wcstod()

5801 In the DESCRIPTION section, change from:

5802 minus-sign, the sequence shall be interpreted as negated.

5803 to:

5804 <hyphen-minus>, the sequence shall be interpreted as negated.

5805 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

5806

5807 **Change Number: XSH/TC2/D4/0392** [796]

5808 On Page: 2224 Line: 69991 Section: wcstod()  
5809 (2013 edition Page: 2246 Line: 71199)

5810 Change from:

5811 In other than the C [CX]or POSIX[/CX] locale[CX]s[/CX], other implementation-defined subject  
5812 sequences may be

5813 to:

5814 In other than the C [CX]or POSIX[/CX] locale, additional locale-specific subject sequence forms may be

5815 *Rationale:* Austin Group Defect Report(s) applied: 796. See <http://austingroupbugs.net/view.php?id=796>.

5816 **Change Number: XSH/TC2/D4/0393** [584]

5817 On Page: 2230 Line: 70196 Section: wcstol()

5818 In the DESCRIPTION section, change from:

5819 minus-sign, the sequence shall be interpreted as negated.

5820 to:

5821 <hyphen-minus>, the sequence shall be interpreted as negated.

5822 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

5823 **Change Number: XSH/TC2/D4/0394** [796]

5824 On Page: 2231 Line: 70199 Section: wcstol()  
5825 (2013 edition Page: 2253 Line: 71409)

5826 Change from:

5827 In other than the C [CX]or POSIX[/CX] locale[CX]s[/CX], other implementation-defined subject  
5828 sequences may be

5829 to:

5830 In other than the C [CX]or POSIX[/CX] locale, additional locale-specific subject sequence forms may be

5831 *Rationale:* Austin Group Defect Report(s) applied: 796. See <http://austingroupbugs.net/view.php?id=796>.

5832 **Change Number: XSH/TC2/D4/0395** [584]

5833 On Page: 2237 Line: 70357 Section: wcstoul()

5834 In the DESCRIPTION section, change from:

5835 minus-sign, the sequence shall be interpreted as negated.

5836 to:

5837 <hyphen-minus>, the sequence shall be interpreted as negated.

5838 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

5839 **Change Number: XSH/TC2/D4/0396** [796]

5840 On Page: 2238 Line: 70360 Section: wstoul()

5841 (2013 edition Page: 2260 Line: 71571)

5842 Change from:

5843 In other than the C [CX] or POSIX[/CX] locale[CX]s[/CX], other implementation-defined subject  
5844 sequences may be

5845 to:

5846 In other than the C [CX] or POSIX[/CX] locale, additional locale-specific subject sequence forms may be

5847 *Rationale*: Austin Group Defect Report(s) applied: 796. See <http://austingroupbugs.net/view.php?id=796>.

5848 **Change Number: XSH/TC2/D4/0397** [608]

5849 On Page: 2258 Line: 71006 Section: wordexp()

5850 In the DESCRIPTION section, change from:

5851 *wordexp()* may write messages to *stderr* if syntax errors are detected while expanding words.

5852 to:

5853 *wordexp()* may write messages to *stderr* if syntax errors are detected while expanding words, unless the  
5854 *stderr* stream has wide orientation in which case the behavior is undefined.

5855 *Rationale*: Austin Group Defect Report(s) applied: 608. See <http://austingroupbugs.net/view.php?id=608>.

5856 **Change Number: XSH/TC2/D4/0398** [704]

5857 On Page: 2258 Line: 71009 Section: wordexp()

5858 (2013 edition Page: 2280 Line: 72230)

5859 In the DESCRIPTION section, add a new paragraph:

5860 The results are unspecified if WRDE\_APPEND and WRDE\_REUSE are both specified.

5861 **Rationale:** Austin Group Defect Report(s) applied: 704. See <http://austingroupbugs.net/view.php?id=704>.  
5862 The standard is clear that wordfree() can be called after any wordexp() call, not just successful ones (the  
5863 text in the fifth paragraph of the DESCRIPTION does not contain the word "successful"). However, the last  
5864 sentence of the first paragraph of the RETURN VALUES section states: "In other cases, they shall not be  
5865 modified.", but it is not clear whether "other cases" here refers to all cases other than when wordexp()  
5866 returns WRDE\_NOSPACE (including success), or just to error cases other than WRDE\_NOSPACE. And,  
5867 neither of these possibilities matches the intended behavior.

5868 **Change Number:** XSH/TC2/D4/0399 [704]

5869 On Page: 2259 Line: 71022 Section: wordexp()  
5870 (2013 edition Page: 2281 Line: 72243)

5871 In the RETURN VALUE section, change from:

5872 In other cases, they shall not be modified.

5873 to:

5874 In other error cases, if the WRDE\_APPEND flag was specified, *pwordexp->we\_wordc* and *pwordexp->we\_wordv*  
5875 shall not be modified.

5876 **Rationale:** Austin Group Defect Report(s) applied: 704. See <http://austingroupbugs.net/view.php?id=704>.

5877 **Change Number:** XSH/TC2/D4/0400 [608]

5878 On Page: 2259 Line: 71038 Section: wordexp()

5879 In the APPLICATION USAGE section, add a new paragraph:

5880 Applications which use wide character output functions with *stderr* should ensure that any calls to  
5881 *wordexp()* do not write to *stderr*, by avoiding use of the WRDE\_SHOWERR flag.

5882 **Rationale:** Austin Group Defect Report(s) applied: 608. See <http://austingroupbugs.net/view.php?id=608>.

5883 **Change Number:** XSH/TC2/D4/0401 [676,710]

5884 On Page: 2266 Line: 71250 Section: write()  
5885 (2013 edition Page: 2288 Line: 72480)

5886 In the ERRORS section, before "The *write()* function shall fail if" insert:

5887 The *pwrite()* function shall fail if:

5888 [EINVAL]

5889 The file is a regular file or block special file, and the offset argument is negative. The file  
5890 offset shall remain unchanged.

5891 [ESPIPE]

5892 The file is incapable of seeking.

5893 On Page: 2266 Line: 71275 Section: write()

5894 In the ERRORS section, delete lines 71275-71277 (the pwrite() "shall fail" errors).

5895 *Rationale:* Austin Group Defect Report(s) applied: 676,710. See  
5896 <http://austingroupbugs.net/view.php?id=676>.  
5897 See <http://austingroupbugs.net/view.php?id=710>.  
5898 This is a layered change on XSH/TC1/D5/0743 [215].

5899 **Change Number: XSH/TC2/D4/0402** [966]

5900 On Page: 2269 Line: 71387 Section: write()  
5901 (2013 edition Page: 2291 Line: 72619)

5902 In the RATIONALE section, change from:

5903 This volume of POSIX.1-2008 does not specify behavior of concurrent writes to a file from multiple  
5904 processes.

5905 to:

5906 This volume of POSIX.1-2008 does not specify the behavior of concurrent writes to a regular file from  
5907 multiple threads, except that each write is atomic (see [xref to 2.9.7]).

5908 *Rationale:* Austin Group Defect Report(s) applied: 966. See <http://austingroupbugs.net/view.php?id=966>.

5909

IECNORM.COM : Click to view the full PDF of ISO/IEC/IEEE 9945:2009/Cor 2:2017

## 5910 4. Changes to Shell and Utilities

5911 This section contains the set of changes to the text of the Shell and Utilities.  
5912 [Note to reviewers: References to defect reports are provided to aid reviewers.]

5913 **Change Number: XCU/TC2/D4/0001** [666]

5914 On Page: 2285 Line: 71853 Section: 1.2 Utility Limits

5915 In the Name column of Table 1-3, change from:

5916 `{_POSIX2_RE_DUP_MAX}`

5917 to:

5918 `{_POSIX_RE_DUP_MAX}`

5919 In the Description column of Table 1-3, change from:

5920 The maximum number of repeated occurrences of a BRE permitted when using the interval notation  
5921  $\{m,n\}$ ; see XBD Section 9.3.6 (on page 186).

5922 to:

5923 Maximum number of repeated occurrences of a BRE or ERE interval expression; see XBD Section 9.3.6  
5924 (on page 186) and XBD Section 9.4.6 (on page 189).

5925 On Page: 2287 Line: 71911 Section: 1.2 Utility Limits

5926 In the Description column of Table 1-4, change from:

5927 The maximum number of repeated occurrences of a BRE permitted when using the interval notation  
5928  $\{m,n\}$ ; see XBD Section 9.3.6 (on page 186).

5929 to:

5930 Maximum number of repeated occurrences of a BRE or ERE interval expression; see XBD Section 9.3.6  
5931 (on page 186) and XBD Section 9.4.6 (on page 189).

5932 In the Minimum Value column of Table 1-4, change from:

5933 `{_POSIX2_RE_DUP_MAX}`

5934 to:

5935 `{_POSIX_RE_DUP_MAX}`

5936 *Rationale:* Austin Group Defect Report(s) applied: 666. See <http://austingroupbugs.net/view.php?id=666>.

5937 **Change Number: XCU/TC2/D4/0002** [584]

5938 On Page: 2289 Line: 71989 Section: 1.4 Utility Description Defaults

5939 Change from:

5940 <hyphen>

5941 to:

5942 <hyphen-minus>

5943 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

5944 **Change Number: XCU/TC2/D4/0003** [913]

5945 On Page: 2293 Line: 72201 Section: 1.4 Utility Description Defaults

5946 (2013 edition Page: 2315 Line: 73439)

5947 Change from:

5948 Utilities may terminate prematurely if they encounter: invalid usage of options, arguments, or environment  
5949 variables; invalid usage of the complex syntaxes expressed in EXTENDED DESCRIPTION sections;  
5950 difficulties accessing, creating, reading, or writing files; or difficulties associated with the privileges of the  
5951 process.

5952 to:

5953 Utilities may terminate prematurely if they encounter: invalid usage of options, arguments, or environment  
5954 variables; invalid usage of the complex syntaxes expressed in EXTENDED DESCRIPTION sections;  
5955 resource exhaustion; difficulties accessing, creating, reading, or writing files; or difficulties associated with  
5956 the privileges of the process.

5957 *Rationale:* Austin Group Defect Report(s) applied: 913. See <http://austingroupbugs.net/view.php?id=913>.

5958 **Change Number: XCU/TC2/D4/0004** [705]

5959 On Page: 2296 Line: 72310 Section: 1.6 Built-In Utilities

5960 (2013 edition Page: 2318 Line: 73457)

5961 In Table 1-5 Regular Built-In Utilities, add *hash*, *type* and *ulimit* to the table.

5962 *Rationale:* Austin Group Defect Report(s) applied: 705. See <http://austingroupbugs.net/view.php?id=705>.

5963 **Change Number: XCU/TC2/D4/0005** [718]

5964 On Page: 2299 Line: 72403 Section: 2.3 Token Recognition

5965 Change from:

5966 The shell shall read its input in terms of lines from a file, from a terminal in the case of an interactive shell,

5967 or from a string in the case of *sh -c* or *system()*.

5968 to:

5969 The shell shall read its input in terms of lines. (For details about how the shell reads its input, see the  
5970 description of the *sh* utility on page xxx.)

5971 *Rationale:* Austin Group Defect Report(s) applied: 718. See <http://austingroupbugs.net/view.php?id=718>.

5972 **Change Number:** XCU/TC2/D4/0006 [647]

5973 On Page: 2299 Line: 72417 Section: 2.3 Token Recognition

5974 In rule 1, change from:

5975 1. If the end of input is recognized, the current token shall be delimited. If there is no current token, the  
5976 end-of-input indicator shall be returned as the token.

5977 to:

5978 1. If the end of input is recognized, the current token (if any) shall be delimited.

5979 *Rationale:* Austin Group Defect Report(s) applied: 647. See <http://austingroupbugs.net/view.php?id=647>.

5980 **Change Number:** XCU/TC2/D4/0007 [568]

5981 On Page: 2299 Line: 72419-72424 Section: 2.3 Token Recognition

5982 In rule 2, on line 72420 change from:

5983 current characters

5984 to:

5985 previous characters

5986 In rule 3, on line 72423 change from:

5987 current characters

5988 to:

5989 previous characters

5990 *Rationale:* Austin Group Defect Report(s) applied: 568. See <http://austingroupbugs.net/view.php?id=568>.

5991

5992 **Change Number: XCU/TC2/D4/0008** [648]

5993 On Page: 2300 Line: 72448 Section: 2.3 Token Recognition

5994 Delete rule 7:

5995 7. If the current character is an unquoted <newline>, the current token shall be delimited.

5996 *Rationale:* Austin Group Defect Report(s) applied: 648. See <http://austingroupbugs.net/view.php?id=648>.

5997 **Change Number: XCU/TC2/D4/0009** [888]

5998 On Page: 2302 Line: 72511-72523 Section: 2.5.2 Special Parameters

5999 (2013 edition Page: 2324 Line: 73742-73754)

6000 Replace the descriptions of the '@' and '\*' special parameters with:

6001 @

6002 Expands to the positional parameters, starting from one, initially producing one field for each  
 6003 positional parameter that is set. When the expansion occurs in a context where field splitting will be  
 6004 performed, any empty fields may be discarded and each of the non-empty fields shall be further split  
 6005 as described in [xref to 2.6.5 Field Splitting]. When the expansion occurs within double-quotes, the  
 6006 behavior is unspecified unless one of the following is true:

- 6007 • Field splitting as described in [xref to 2.6.5 Field Splitting] would be performed if the  
 6008 expansion were not within double-quotes (regardless of whether field splitting would have  
 6009 any effect, for example if IFS is null).
- 6010 • The double-quotes are within the *word* of a \${parameter:-word} or a \${parameter:+word}  
 6011 expansion (with or without the <colon>; see [xref to 2.6.2 Parameter Expansion]) which  
 6012 would have been subject to field splitting if *parameter* had been expanded instead of *word*.

6013 If one of these conditions is true, the initial fields shall be retained as separate fields except that, if  
 6014 the parameter being expanded was embedded within a word, the first field shall be joined with the  
 6015 beginning part of the original word and the last field shall be joined with the end part of the original  
 6016 word. In all other contexts the results of the expansion are unspecified. If there are no positional  
 6017 parameters, the expansion of '@' shall generate zero fields, even when '@' is within double-quotes;  
 6018 however, if the expansion is embedded within a word which contains one or more other parts that  
 6019 expand to a quoted null string, these null string(s) shall still produce an empty field, except that if  
 6020 the other parts are all within the same double-quotes as the '@', it is unspecified whether the result is  
 6021 zero fields or one empty field.

6022 \*

6023 Expands to the positional parameters, starting from one, initially producing one field for each  
 6024 positional parameter that is set. When the expansion occurs in a context where field splitting will be  
 6025 performed, any empty fields may be discarded and each of the non-empty fields shall be further split  
 6026 as described in [xref to 2.6.5 Field Splitting]. When the expansion occurs in a context where field  
 6027 splitting will not be performed, the initial fields shall be joined to form a single field with the value  
 6028 of each parameter separated by the first character of the IFS variable if IFS contains at least one  
 6029 character, or separated by a <space> if IFS is unset, or with no separation if IFS is set to a null  
 6030 string.

6031 *Rationale:* Austin Group Defect Report(s) applied: 888. See <http://austingroupbugs.net/view.php?id=888>.

6032 **Change Number: XCU/TC2/D4/0010 [888]**

6033 On Page: 2303 Line: 72562 Section: 2.5.3 Shell Variables  
6034 (2013 edition Page: 2325 Line: 73791)

6035 In the description of IFS, change from:

6036 A string treated as a list of characters that is used for field splitting and to split lines into fields with the  
6037 *read* command.

6038 If *IFS* is not set, it shall behave as normal for an unset variable, except that field splitting by the shell and  
6039 line splitting by the *read* command shall ...

6040 to:

6041 A string treated as a list of characters that is used for field splitting, expansion of the '\*' special parameter,  
6042 and to split lines into fields with the *read* utility. If the value of *IFS* includes any bytes that do not form part  
6043 of a valid character, the results of field splitting, expansion of '\*', and use of the *read* utility are unspecified.

6044 If *IFS* is not set, it shall behave as normal for an unset variable, except that field splitting by the shell and  
6045 line splitting by the *read* utility shall ...

6046 *Rationale*: Austin Group Defect Report(s) applied: 888. See <http://austingroupbugs.net/view.php?id=888>.

6047 **Change Number: XCU/TC2/D4/0011 [884]**

6048 On Page: 2303 Line: 72568 Section: 2.5.3 Shell Variables  
6049 (2013 edition Page: 2325 Line: 73796)

6050 In the description of IFS, change from:

6051 Implementations may ignore the value of *IFS* in the environment, or the absence of *IFS* from the  
6052 environment, at the time the shell is invoked, in which case the shell shall set *IFS* to  
6053 <space><tab><newline> when it is invoked.

6054 to:

6055 The shell shall set *IFS* to <space><tab><newline> when it is invoked.

6056 *Rationale*: Austin Group Defect Report(s) applied: 884. See <http://austingroupbugs.net/view.php?id=884>.  
6057 Some old shells did inherit IFS from the environment, but since most shell scripts do not set IFS as one of  
6058 the steps in their initialization, this creates a security hole. Most, if not all, recent shells initialize IFS when  
6059 the shell is invoked and do not change IFS in a subshell environment. This is the desired behavior.

6060 **Change Number: XCU/TC2/D4/0012 [494]**

6061 On Page: 2304 Line: 72602 Section: 2.5.3 Shell Variables

6062 Delete the following from the description of the PPID shell variable:

6063 This volume of POSIX.1-2008 specifies the effects of the variable only for systems supporting the User

6064 Portability Utilities option.

6065 *Rationale*: Austin Group Defect Report(s) applied: 494. See <http://austingroupbugs.net/view.php?id=494>.

6066 **Change Number: XCU/TC2/D4/0013** [888]

6067 On Page: 2306 Line: 72709 Section: 2.6.2 Parameter Expansion  
 6068 (2013 edition Page: 2328 Line: 73940)

6069 Change from:

6070 Field splitting shall not be performed on the results of the expansion, with the exception of '@'; see [xref to  
 6071 2.5.2].

6072 to:

6073 Field splitting shall not be performed on the results of the expansion.

6074 *Rationale*: Austin Group Defect Report(s) applied: 888. See <http://austingroupbugs.net/view.php?id=888>.

6075 **Change Number: XCU/TC2/D4/0014** [867]

6076 On Page: 2306 Line: 72716 Section: 2.6.2 Parameter Expansion  
 6077 (2013 edition Page: 2328 Line: 73947)

6078 Delete the sentence:

6079 (For example,  `${foo-bar}xyz` would result in the expansion of **foo** followed by the string **xyz** if **foo** is  
 6080 set, else the string "barxyz" ).

6081 On Page: 2308 Line: 72769 Section: 2.6.2 Parameter Expansion  
 6082 (2013 edition Page: 2330 Line: 74014)

6083 Insert the following before the  `${parameter:-word}` example:

6084  `${parameter-word}`

6085 This example demonstrates the difference between unset and set to the empty string, as well  
 6086 as the rules for finding the delimiting close brace.

6087 `foo=asdf`  
 6088 `echo ${foo-bar}xyz`  
 6089 `asdfxyz`  
 6090 `foo=`  
 6091 `echo ${foo-bar}xyz`  
 6092 `xyz`  
 6093 `unset foo`  
 6094 `echo ${foo-bar}xyz`  
 6095 `barxyz`

6096 *Rationale*: Austin Group Defect Report(s) applied: 867. See <http://austingroupbugs.net/view.php?id=867>.

6097 **Change Number: XCU/TC2/D4/0015** [584]

6098 On Page: 2310 Line: 72861 Section: Section 2.6.4 Arithmetic Expansion  
6099 (2013 edition Page: 2332 Line: 74115)

6100 Change from:

6101 If the shell variable *x* contains a value that forms a valid integer constant, then ...

6102 to:

6103 If the shell variable *x* contains a value that forms a valid integer constant, optionally including a leading  
6104 <plus-sign> or <hyphen-minus>, then ...

6105 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.  
6106 This item is a layered change on XCU/TC1/D5/0020 [50] applied in TC1, the difference being a change  
6107 from:

6108 leading plus or minus sign

6109 to:

6110 leading <plus-sign> or <hyphen-minus>

6111 **Change Number: XCU/TC2/D4/0016** [832]

6112 On Page: 2311 Line: 72882 Section: Section 2.6.5 Field Splitting  
6113 (2013 edition Page: 2333 Line: 74136)

6114 Change from:

6115 The shell shall treat each character of the *IFS* as a delimiter and use the delimiters as field terminators to  
6116 split the results of parameter expansion and command substitution into fields.

6117 to:

6118 The shell shall treat each character of the *IFS* as a delimiter and use the delimiters as field terminators to  
6119 split the results of parameter expansion, command substitution, and arithmetic expansion into fields.

6120 *Rationale*: Austin Group Defect Report(s) applied: 832. See <http://austingroupbugs.net/view.php?id=832>.  
6121 Section 2.6.5 doesn't mention arithmetic expansion (which specifies when field splitting is done), but  
6122 section 2.6 (where it talks about the order of word expansions) does require field splitting after arithmetic  
6123 expansions.

6124 **Change Number: XCU/TC2/D4/0017** [890]

6125 On Page: 2313 Line: 72971 Section: 2.7.4 Here-Document  
6126 (2013 edition Page: 2335 Line: 74224)

6127 Change from:

6128 The redirection operators "<<" and "<<-" both allow redirection of lines contained in a shell input file,  
6129 known as a "here-document", to the input of a command.

6130 to:

6131 The redirection operators "<<" and "<<-" both allow redirection of subsequent lines read by the shell to the  
6132 input of a command. The redirected lines are known as a "here-document".

6133 On Page: 2313 Line: 72981 Section: 2.7.4 Here-Document  
6134 (2013 edition Page: 2335 Line: 74234)

6135 After:

6136 where the optional *n* represents the file descriptor number. If the number is omitted, the here-document  
6137 refers to standard input (file descriptor 0).

6138 add a new sentence:

6139 It is unspecified whether the file descriptor is opened as a regular file, a special file, or a pipe. Portable  
6140 applications cannot rely on the file descriptor being seekable (see [xref to XSH lseek()]).

6141 *Rationale*: Austin Group Defect Report(s) applied: 890. See <http://austingroupbugs.net/view.php?id=890>.

6142 **Change Number: XCU/TC2/D4/0018** [583]

6143 On Page: 2313 Line: 72982,72985 Section: 2.7.4 Here-Document

6144 On line 72982 change from:

6145 If any character in *word*

6146 to:

6147 If any part of *word*

6148 On line 72985 change from:

6149 If no characters in *word* are

6150 to:

6151 If no part of *word* is

6152 *Rationale*: Austin Group Defect Report(s) applied: 583. See <http://austingroupbugs.net/view.php?id=583>.

6153 **Change Number: XCU/TC2/D4/0019** [580]

6154 On Page: 2314 Line: 72990 Section: 2.7.4 Here-Document

6155 Change from:

6156 redirection symbol

6157 to:

6158 redirection operator

6159 *Rationale:* Austin Group Defect Report(s) applied: 580. See <http://austingroupbugs.net/view.php?id=580>.

6160 **Change Number: XCU/TC2/D4/0020** [882]

6161 On Page: 2315 Line: 73032-73042 Section: 2.8.1 Consequences of Shell Errors  
 6162 (2013 edition Page: 2337 Line: 74285-74295)

6163 Change from:

6164 For a non-interactive shell, an error condition encountered by a special built-in (see Section 2.14) or other  
 6165 type of utility shall cause the shell to write a diagnostic message to standard error and exit as shown in the  
 6166 following table:

Error	Special Built-In	Other Utilities
Shell language syntax error	Shall exit	Shall exit
Utility syntax error (option or operand error)	Shall exit	Shall not exit
Redirection error	Shall exit	Shall not exit
Variable assignment error	Shall exit	Shall not exit
Expansion error	Shall exit	Shall exit
Command not found	N/A	May exit
Dot script not found	Shall exit	N/A

6167

6168 to:

6169 Certain errors shall cause the shell to write a diagnostic message to standard error and exit as shown in the  
 6170 following table:

Error	Non-Interactive Shell	Interactive Shell	Shell Diagnostic Message Required
Shell language syntax error	shall exit	shall not exit	yes
Special built-in utility error	shall exit	shall not exit	no <sup>1</sup>
Other utility (not a special built-in) error	shall not exit	shall not exit	no <sup>2</sup>

Redirection error with special built-in utilities	shall exit	shall not exit	yes
Redirection error with compound commands	may exit <sup>3</sup>	shall not exit	yes
Redirection error with function execution	may exit <sup>3</sup>	shall not exit	yes
Redirection error with other utilities (not special built-ins)	shall not exit	shall not exit	yes
Variable assignment error	shall exit	shall not exit	yes
Expansion error	shall exit	shall not exit	yes
Command not found	may exit	shall not exit	yes

6171

6172 Notes:

6173 1. Although special built-ins are part of the shell, a diagnostic message written by a special built-in is  
 6174 not considered to be a shell diagnostic message, and can be redirected like any other utility.  
 6175 2. The shell is not required to write a diagnostic message, but the utility itself shall write a diagnostic  
 6176 message if required to do so.  
 6177 3. A future version of this standard may require the shell to not exit in this condition.

6178 *Rationale:* Austin Group Defect Report(s) applied: 882. See <http://austingroupbugs.net/view.php?id=882>.  
 6179 The standard is unclear on what constitutes a "Shell Error".

6180 **Change Number: XCU/TC2/D4/0021** [717,882]

6181 On Page: 2315 Line: 73047-73051 Section: 2.8.1 Consequences of Shell Errors  
 6182 (2013 edition Page: 233 Line: 74300-74304)

6183 Change from:

6184 If any of the errors shown as "shall exit" or "(may) exit" occur in a subshell, the subshell shall (respectively  
 6185 may) exit with a non-zero status, but the script containing the subshell shall not exit because of the error.

6186 In all of the cases shown in the table, an interactive shell shall write a diagnostic message to standard error  
 6187 without exiting.

6188 to:

6189 If any of the errors shown as "shall exit" or "may exit" occur in a subshell environment, the shell shall  
 6190 (respectively may) exit from the subshell environment with a non-zero status and continue in the  
 6191 environment from which that subshell environment was invoked.

6192 In all of the cases shown in the table where an interactive shell is required not to exit, the shell shall not  
6193 perform any further processing of the command in which the error occurred.

6194 *Rationale*: Austin Group Defect Report(s) applied: 717,882. See  
6195 <http://austingroupbugs.net/view.php?id=717>.  
6196 See <http://austingroupbugs.net/view.php?id=882>.  
6197 The changes made here make the standard reflect existing practice.

6198 **Change Number: XCU/TC2/D4/0022** [717]

6199 On Page: 2315 Line: 73059 Section: 2.8.2 Exit Status for Commands

6200 Change from:

6201 If a command fails during word expansion or redirection, its exit status shall be greater than zero.

6202 to:

6203 If a command fails during word expansion or redirection, its exit status shall be between 1 and 125  
6204 inclusive.

6205 *Rationale*: Austin Group Defect Report(s) applied: 717. See <http://austingroupbugs.net/view.php?id=717>.  
6206 The changes made here make the standard reflect existing practice.

6207 **Change Number: XCU/TC2/D4/0023** [473]

6208 On Page: 2316 Line: 73070 Section: 2.9 Shell Commands

6209 Change from:

6210 ... represent the syntax.

6211 to:

6212 ... represent the syntax. In particular the representations include spacing between tokens in some places  
6213 where blanks would not be necessary (when one of the tokens is an operator).

6214 *Rationale*: Austin Group Defect Report(s) applied: 473. See <http://austingroupbugs.net/view.php?id=473>.

6215 **Change Number: XCU/TC2/D4/0024** [654]

6216 On Page: 2317 Line: 73101 Section: 2.9.1 Simple Commands

6217 Change from:

6218 If no command name results, variable assignments shall affect the current execution environment.  
6219 Otherwise, the variable assignments shall be exported for the execution environment of the command and  
6220 shall not affect the current execution environment (except for special built-ins). If any of the variable  
6221 assignments attempt to assign a value to a read-only variable, a variable assignment error shall occur.

6222 to:

6223 Variable assignments shall be performed as follows:

6224     • If no command name results, variable assignments shall affect the current execution environment.

6225     • If the command name is not a special built-in utility or function, the variable assignments shall be  
 6226        exported for the execution environment of the command and shall not affect the current execution  
 6227        environment except as a side-effect of the expansions performed in step 4. In this case it is  
 6228        unspecified:

6229        — Whether or not the assignments are visible for subsequent expansions in step 4

6230        — Whether variable assignments made as side-effects of these expansions are visible for  
 6231        subsequent expansions in step 4, or in the current shell execution environment, or both

6232     • If the command name is a standard utility implemented as a function (see XBD Section 4.21), the  
 6233        effect of variable assignments shall be as if the utility was not implemented as a function.

6234     • If the command name is a special built-in utility, variable assignments shall affect the current  
 6235        execution environment. Unless the *set -a* option is on (see the *set* special built-in utility in Section  
 6236        2.14), it is unspecified:

6237        — Whether or not the variables gain the *export* attribute during the execution of the special  
 6238        built-in utility

6239        — Whether or not *export* attributes gained as a result of the variable assignments persist after  
 6240        the completion of the special built-in utility

6241     • If the command name is a function that is not a standard utility implemented as a function, variable  
 6242        assignments shall affect the current execution environment during the execution of the function. It  
 6243        is unspecified:

6244        — Whether or not the variable assignments persist after the completion of the function

6245        — Whether or not the variables gain the *export* attribute during the execution of the function

6246        — Whether or not *export* attributes gained as a result of the variable assignments persist after  
 6247        the completion of the function (if variable assignments persist after the completion of the  
 6248        function)

6249     If any of the variable assignments attempt to assign a value to a variable for which the *readonly* attribute is  
 6250        set in the current shell environment (regardless of whether the assignment is made in that environment), a  
 6251        variable assignment error shall occur. See [xref to 2.8.1] for the consequences of these errors.

6252     *Rationale:* Austin Group Defect Report(s) applied: 654. See <http://austingroupbugs.net/view.php?id=654>.  
 6253     This change is layered on Change Number XCU/TC1/D5/0021 [255] in TC1.

6254     **Change Number: XCU/TC2/D4/0025** [935]

6255     On Page: 2317 Line: 73124 Section: 2.9.1.1 Command Search and Execution  
 6256        (2013 edition Page: 2339 Line: 74383)

6257     In list item 1. insert a new nested item:

6258     b. If the command name matches the name of a utility listed in the following table, the results are  
6259     unspecified:

6260       *alloc*  
6261       *autoload*  
6262       *bind*  
6263       *bindkey*  
6264       *builtin*  
6265       *bye*  
6266       *caller*  
6267       *cap*  
6268       *chdir*  
6269       *clone*  
6270       *comparguments*  
6271       *compcall*  
6272       *compcctl*  
6273       *compdescribe*  
6274       *compfiles*  
6275       *compgen*  
6276       *compgroups*  
6277       *complete*  
6278       *compquote*  
6279       *comptags*  
6280       *comptrry*  
6281       *compvalues*  
6282       *declare*  
6283       *dirs*  
6284       *disable*  
6285       *disown*  
6286       *dosh*  
6287       *echotc*  
6288       *echoti*  
6289       *help*  
6290       *history*  
6291       *hist*  
6292       *let*  
6293       *local*  
6294       *login*  
6295       *logout*  
6296       *map*  
6297       *mapfile*  
6298       *popd*  
6299       *print*  
6300       *pushd*  
6301       *readarray*  
6302       *repeat*  
6303       *savehistory*  
6304       *source*  
6305       *shopt*  
6306       *stop*  
6307       *suspend*  
6308       *typeset*  
6309       *whence*

6310     and renumber 1.b, c, and d to 1.c, d, and e.

6311 *Rationale*: Austin Group Defect Report(s) applied: 935. See <http://austingroupbugs.net/view.php?id=935>.

6312 **Change Number: XCU/TC2/D4/0026** [705]

6313 On Page: 2317 Line: 73130 Section: 2.9.1.1 Command Search and Execution  
 6314 (2013 edition Page: 2339 Line: 74388)

6315 In item 1.c, change from:

6316 If the command name matches the name of a utility listed in the following table, that utility shall be  
 6317 invoked.

6318 to:

6319 If the command name matches the name [XSI] of the *type* or *ulimit* utility, or/[XSI] of a utility listed in the  
 6320 following table, that utility shall be invoked.

6321 and add *hash* to the table.

6322 *Rationale*: Austin Group Defect Report(s) applied: 705. See <http://austingroupbugs.net/view.php?id=705>.

6323 **Change Number: XCU/TC2/D4/0027** [521]

6324 On Page: 2318 Line: 73179 Section: 2.9.2 Pipelines

6325 After:

6326 `[!] command1 [ | command2 ...]`

6327 add a new paragraph:

6328 If the pipeline begins with the reserved word `!` and *command1* is a subshell command, the application shall  
 6329 ensure that the `(` operator at the beginning of *command1* is separated from the `!` by one or more `<blank>`  
 6330 characters. The behavior of the reserved word `!` immediately followed by the `(` operator is unspecified.

6331 *Rationale*: Austin Group Defect Report(s) applied: 521. See <http://austingroupbugs.net/view.php?id=521>.  
 6332 Requiring a negated subshell conflicts with shells that provide an extension of negated pathname  
 6333 expansion.

6334 **Change Number: XCU/TC2/D4/0028** [760]

6335 On Page: 2320 Line: 73226 Section: 2.9.3.1 Asynchronous Lists

6336 Change from:

6337 The standard input for an asynchronous list, before any explicit redirections are performed, shall be  
 6338 considered to be assigned to a file that has the same properties as `/dev/null`. If it is an interactive shell, this  
 6339 need not happen.

6340 to:

6341 If job control is disabled (see *set -m*), the standard input for an asynchronous list, before any explicit  
6342 redirections are performed, shall be considered to be assigned to a file that has the same properties as  
6343 */dev/null*. This shall not happen if job control is enabled.

6344 *Rationale*: Austin Group Defect Report(s) applied: 760. See <http://austingroupbugs.net/view.php?id=760>.

6345 **Change Number: XCU/TC2/D4/0029** [473]

6346 On Page: 2321 Line: 73274 Section: 2.9.1.4 Grouping Commands

6347 Change from:

6348 *(compound-list)*

6349 to:

6350 *( compound-list )*

6351 On Page: 2321 Line: 73277 Section: 2.9.4.1 Grouping Commands

6352 Change from:

6353 *{ compound-list; }*

6354 to:

6355 *{ compound-list ; }*

6356 On Page: 2322 Line: 73312 Section: 2.9.4.3 Case Conditional Construct

6357 Change from:

6358 *case word in*  
6359 *[ ( ] pattern1 ) compound-list ; ;*  
6360 *[ [ ( ] pattern [ | pattern ] ... ) compound-list ; ; ] ...*  
6361 *[ [ ( ] pattern [ | pattern ] ... ) compound-list ]*  
6362 *esac*

6363 to:

6364 *case word in*  
6365 *[ ( ] pattern1 ) compound-list ; ;*  
6366 *[ [ ( ] pattern [ | pattern ] ... ) compound-list ; ; ] ...*  
6367 *[ [ ( ] pattern [ | pattern ] ... ) compound-list ]*  
6368 *esac*

6369 On Page: 2324 Line: 73379 Section: 2.9.5 Function Definition Command

6370 Change from:

6371 *fname() compound-command[io-redirect ...]*

6372 to:

6373 *fname* ( ) *compound-command* [*io-redirect* ...]

6374 *Rationale*: Austin Group Defect Report(s) applied: 473. See <http://austingroupbugs.net/view.php?id=473>.

6375 **Change Number: XCU/TC2/D4/0030** [654]

6376 On Page: 2324 Line: 73391 Section: 2.9.5 Function Definition Command  
 6377 (2013 edition Page: 2347 Line: 74675-74676)

6378 Change from:

6379 When a function is executed, it shall have the syntax-error and variable-assignment properties described for  
 6380 special built-in utilities in the enumerated list at the beginning of Section 2.14.

6381 to:

6382 When a function is executed, it shall have the syntax-error properties described for special built-in utilities  
 6383 in the first item in the enumerated list at the beginning of Section 2.14.

6384 *Rationale*: Austin Group Defect Report(s) applied: 654. See <http://austingroupbugs.net/view.php?id=654>.

6385 **Change Number: XCU/TC2/D4/0031** [648]

6386 On Page: 2325 Line: 73415 Section: 2.10.1 Shell Grammar Lexical Conventions

6387 Delete rule 1:

6388 1. A <newline> shall be returned as the token identifier NEWLINE.

6389 *Rationale*: Austin Group Defect Report(s) applied: 648. See <http://austingroupbugs.net/view.php?id=648>.

6390 **Change Number: XCU/TC2/D4/0032** [574,646]

6391 On Page: 2325 Line: 73421 Section: 2.10.1 Shell Grammar Lexical Conventions

6392 Change from:

6393 **ASSIGNMENT**

6394 to:

6395 **ASSIGNMENT\_WORD**

6396 *Rationale*: Austin Group Defect Report(s) applied: 574,646. See

6397 <http://austingroupbugs.net/view.php?id=574>.

6398 See <http://austingroupbugs.net/view.php?id=646>.

6399

6400   **Change Number: XCU/TC2/D4/0033** [643,839]

6401   On Page: 2326 Line: 73475-73481 Section: 2.10.2 Shell Grammar Rules  
6402   (2013 edition Page: 2349 Line: 74755-74761)

6403   Change rule 7.b from:

6404   If the **TOKEN** contains the <equals-sign> character:

6405       

- If it begins with '=', the token **WORD** shall be returned.

6406       

- If all the characters preceding '=' form a valid name (see XBD Section 3.230, on page 70), the token **ASSIGNMENT\_WORD** shall be returned. (Quoted characters cannot participate in forming a valid name.)

6409       

- Otherwise, it is unspecified whether it is **ASSIGNMENT\_WORD** or **WORD** that is returned.

6410   to:

6411   If the **TOKEN** contains an unquoted (as determined while applying rule #4 from section 2.3) <equals-sign> character that is not part of an embedded parameter expansion, command substitution, or arithmetic expansion construct (as determined while applying rule #5 from section 2.3):

6414       

- If the **TOKEN** begins with '=', then rule 1 shall be applied.

6415       

- If all the characters in the **TOKEN** preceding the first such <equals-sign> form a valid name (see XBD Section 3.230, on page 70), the token **ASSIGNMENT\_WORD** shall be returned.

6417       

- Otherwise, it is unspecified whether rule 1 is applied or **ASSIGNMENT\_WORD** is returned.

6418   Otherwise, rule 1 shall be applied.

6419   *Rationale:* Austin Group Defect Report(s) applied: 643,839. See  
6420   <http://austingroupbugs.net/view.php?id=643>.  
6421   See <http://austingroupbugs.net/view.php?id=839>.

6422   **Change Number: XCU/TC2/D4/0034** [643]

6423   On Page: 2326 Line: 73482 Section: 2.10.2 Shell Grammar Rules

6424   In rule 7, change from:

6425   Assignment to the **NAME** shall occur ...

6426   to:

6427   Assignment to the name within a returned **ASSIGNMENT\_WORD** token shall occur ...

6428   *Rationale:* Austin Group Defect Report(s) applied: 643. See <http://austingroupbugs.net/view.php?id=643>.

6429

6430 **Change Number: XCU/TC2/D4/0035** [648]

6431 On Page: 2327 Line: 73500 Section: 2.10.2 Shell Grammar Rules

6432 In the shell grammar, change from:

6433 /\* The following are the operators mentioned above \*/

6434 to:

6435 /\* The following are the operators (see XBD 3.255) containing more  
 6436 than one character \*/

6437 *Rationale*: Austin Group Defect Report(s) applied: 648. See <http://austingroupbugs.net/view.php?id=648>.

6438 **Change Number: XCU/TC2/D4/0036** [736]

6439 On Page: 2327 Line: 73521 Section: 2.10.2 Shell Grammar Rules

6440 (2013 edition Page: 2350 Line: 74801-74808)

6441 In the shell grammar, change from:

6442 %start complete\_command  
 6443 %%  
 6444 complete\_command : list separator  
 6445 | list  
 6446 ;

6447 to:

6448 %start program  
 6449 %%  
 6450 program : linebreak complete\_commands linebreak  
 6451 | linebreak  
 6452 ;  
 6453 complete\_commands: complete\_commands newline\_list complete\_command  
 6454 | complete\_command  
 6455 ;  
 6456 complete\_command : list separator\_op  
 6457 | list  
 6458 ;

6459 *Rationale*: Austin Group Defect Report(s) applied: 736. See <http://austingroupbugs.net/view.php?id=736>.

6460 **Change Number: XCU/TC2/D4/0037** [737]

6461 On Page: 2328 Line: 73554 Section: 2.10.2 Shell Grammar Rules

6462 (2013 edition Page: 2350 Line: 74834-74838)

6463 In the shell grammar, change from:

6464 compound\_list : term  
 6465 | newline\_list term  
 6466 | term separator

6467 | newline\_list term separator  
 6468 ;

6469 to:

6470 compound\_list : linebreak term  
 6471 | linebreak term separator  
 6472 ;

6473 *Rationale*: Austin Group Defect Report(s) applied: 737. See <http://austingroupbugs.net/view.php?id=737>

6474 **Change Number: XCU/TC2/D4/0038** [581]

6475 On Page: 2328 Line: 73562-73564 Section: 2.10.2 Shell Grammar Rules

6476 In the shell grammar, change from:

6477 for\_clause : For name linebreak do\_group  
 6478 | For name linebreak in sequential\_sep do\_group  
 6479 | For name linebreak in wordlist sequential\_sep do\_group  
 6480 ;

6481 to:

6482 for\_clause : For name do\_group  
 6483 | For name sequential\_sep do\_group  
 6484 | For name linebreak in sequential\_sep do\_group  
 6485 | For name linebreak in wordlist sequential\_sep do\_group  
 6486 ;

6487 *Rationale*: Austin Group Defect Report(s) applied: 581. See <http://austingroupbugs.net/view.php?id=581>.

6488 **Change Number: XCU/TC2/D4/0039** [735]

6489 On Page: 2328 Line: 73583 Section: 2.10.2 Shell Grammar Rules  
 6490 (2013 edition Page: 2351 Line: 74863-74866)

6491 In the shell grammar, change from:

6492 case\_item\_ns : pattern ')' linebreak  
 6493 | pattern ')' compound\_list linebreak  
 6494 | '(' pattern ')' linebreak  
 6495 | '(' pattern ')' compound\_list linebreak  
 6496 ;

6497 to:

6498 case\_item\_ns : pattern ')' linebreak  
 6499 | pattern ')' compound\_list  
 6500 | '(' pattern ')' linebreak  
 6501 | '(' pattern ')' compound\_list  
 6502 ;

6503 *Rationale*: Austin Group Defect Report(s) applied: 735. See <http://austingroupbugs.net/view.php?id=735>.

6504 **Change Number: XCU/TC2/D4/0040** [750]

6505 On Page: 2330 Line: 73676 Section: 2.11 Signals and Error Handling  
 6506 (2013 edition Page: 2353 Line: 74957)

6507 Change from:

6508 When a command is in an asynchronous list, it shall inherit from the shell a signal action of ignored  
 6509 (SIG\_IGN) for the SIGQUIT and SIGINT signals, and may inherit a signal mask in which SIGQUIT and  
 6510 SIGINT are blocked. Otherwise, the signal actions and signal mask inherited by the command shall be the  
 6511 same as ...

6512 to:

6513 If job control is disabled (see the description of *set -m*) when the shell executes an asynchronous list, the  
 6514 commands in the list shall inherit from the shell a signal action of ignored (SIG\_IGN) for the SIGINT and  
 6515 SIGQUIT signals. In all other cases, commands executed by the shell shall inherit the same signal actions  
 6516 as ...

6517 *Rationale:* Austin Group Defect Report(s) applied: 750. See <http://austingroupbugs.net/view.php?id=750>.

6518 **Change Number: XCU/TC2/D4/0041** [706]

6519 On Page: 2331 Line: 73693 Section: 2.12 Shell Execution Environment  
 6520 (2013 edition Page: 2353 Line: 74974)

6521 In the shell execution environment list, add a new entry with XSI shading:

- File size limit as set by *ulimit*.

6523 *Rationale:* Austin Group Defect Report(s) applied: 706. See <http://austingroupbugs.net/view.php?id=706>.

6524 **Change Number: XCU/TC2/D4/0042** [806]

6525 On Page: 2332 Line: 73736 Section: 2.13.1 Patterns Matching a Single Character  
 6526 (2013 edition Page: 2354 Line: 75017)

6527 Change from:

6528 The escaping <backslash> shall be discarded.

6529 to:

6530 The escaping <backslash> shall be discarded. If a pattern ends with an unescaped <backslash>, it is  
 6531 unspecified whether the pattern does not match anything or the pattern is treated as invalid.

6532 *Rationale:* Austin Group Defect Report(s) applied: 806. See <http://austingroupbugs.net/view.php?id=806>.  
 6533 Existing practice is for a pattern ending in an unescaped <backslash> not to match anything, but it would  
 6534 be useful to allow implementations to diagnose this as user error.

6535

6536 **Change Number: XCU/TC2/D4/0043** [963]

6537 On Page: 2333 Line: 73803 Section: 2.13.3 Patterns Used for Filename Expansion  
6538 (2013 edition Page: 2356 Line: 75082)

6539 Change from:

6540 ... sorted according to the collating sequence in effect in the current locale.

6541 to:

6542 ... sorted according to the collating sequence in effect in the current locale. If this collating sequence does  
6543 not have a total ordering of all characters (see [xref to XBD 7.3.2]), any filenames or pathnames that collate  
6544 equally should be further compared byte-by-byte using the collating sequence for the POSIX locale.

6545 <small>Note: a future version of this standard may require the byte-by-byte further comparison described  
6546 above.</small>

6547 *Rationale:* Austin Group Defect Report(s) applied: 963. See <http://austingroupbugs.net/view.php?id=963>.

6548 **Change Number: XCU/TC2/D4/0044** [882]

6549 On Page: 2334 Line: 73821 Section: 2.14 Special Built-In Utilities  
6550 (2013 edition Page: 2356 Line: 75100)

6551 In list item 1, change from:

6552 A syntax error in a special built-in utility may cause a shell executing that utility to abort, while a syntax  
6553 error in a regular built-in utility shall not cause a shell executing that utility to abort. (See Section 2.8.1 for  
6554 the consequences of errors on interactive and non-interactive shells.) If a special built-in utility  
6555 encountering a syntax error does not abort the shell, its exit value shall be non-zero.

6556 to:

6557 An error in a special built-in utility may cause a shell executing that utility to abort, while an error in a  
6558 regular built-in utility shall not cause a shell executing that utility to abort. (See Section 2.8.1 for the  
6559 consequences of errors on interactive and non-interactive shells.) If a special built-in utility encountering an  
6560 error does not abort the shell, its exit value shall be non-zero.

6561 *Rationale:* Austin Group Defect Report(s) applied: 882. See <http://austingroupbugs.net/view.php?id=882>.

6562 **Change Number: XCU/TC2/D4/0045** [654]

6563 On Page: 2334 Line: 73826 Section: 2.14 Special Built-In Utilities  
6564 (2013 edition Page: 2356 Line: 75105)

6565 In list item 2, change from:

6566 Variable assignments specified with special built-in utilities remain in effect after the built-in completes;  
6567 this shall not be the case with a regular built-in or other utility.

6568 to:

6569 As described in Section 2.9.1, variable assignments preceding the invocation of a special built-in utility  
 6570 remain in effect after the built-in completes; this shall not be the case with a regular built-in or other utility.

6571 *Rationale:* Austin Group Defect Report(s) applied: 654. See <http://austingroupbugs.net/view.php?id=654>.

6572 **Change Number: XCU/TC2/D4/0046** [842]

6573 On Page: 2335 Line: 73839 Section: 2.14 break  
 6574 (2013 edition Page: 2358 Line: 75117)

6575 Change the DESCRIPTION section from:

6576 The *break* utility shall exit from the smallest enclosing **for**, **while**, or **until** loop, if any; or from the *n*th  
 6577 enclosing loop if *n* is specified. The value of *n* is an unsigned decimal integer greater than or equal to 1.  
 6578 The default shall be equivalent to *n*=1. If *n* is greater than the number of enclosing loops, the outermost  
 6579 enclosing loop shall be exited. Execution shall continue with the command immediately following the loop.

6580 to:

6581 If *n* is specified, the *break* utility shall exit from the *n*th enclosing **for**, **while**, or **until** loop. If *n* is not  
 6582 specified, *break* shall behave as if *n* was specified as 1. Execution shall continue with the command  
 6583 immediately following the exited loop. The value of *n* is a positive decimal integer. If *n* is greater than the  
 6584 number of enclosing loops, the outermost enclosing loop shall be exited. If there is no enclosing loop, the  
 6585 behavior is unspecified.

6586 A loop shall enclose a *break* or *continue* command if the loop lexically encloses the command. A loop  
 6587 lexically encloses a *break* or *continue* command if the command is:

- 6588 • executing in the same execution environment (see section 2.12) as the compound-list of the loop's  
 6589 do-group (see section 2.10.2), and
- 6590 • contained in a compound-list associated with the loop (either in the compound-list of the loop's do-  
 6591 group or, if the loop is a **while** or **until** loop, in the compound-list following the **while** or **until**  
 6592 reserved word), and
- 6593 • not in the body of a function whose function definition command (see section 2.9.5) is contained  
 6594 in a compound-list associated with the loop.

6595 If *n* is greater than the number of lexically enclosing loops and there is a non-lexically enclosing loop in  
 6596 progress in the same execution environment as the *break* or *continue* command, it is unspecified whether  
 6597 that loop encloses the command.

6598 On Page: 2336 Line: 73877 Section: 2.14 break  
 6599 (2013 edition Page: 2359 Line: 75155)

6600 In the EXAMPLES section, add to the end of the section:

6601 The results of running the following example are unspecified: There are two loops in progress when the  
 6602 *break* command is executed, and they are in the same execution environment, but neither loop is lexically  
 6603 enclosing the *break* command. (There are no loops lexically enclosing the *continue* commands, either.)

6604 `foo( ) {`

```

6605      for j in 1 2; do
6606          echo 'break 2' >/tmp/do_break
6607          echo " sourcing /tmp/do_break ($j)...""
6608          # the behavior of the break from running the following command
6609          # results in unspecified behavior:
6610          . /tmp/do_break
6611
6612          do_continue() { continue 2; }
6613          echo " running do_continue ($j)...""
6614          # the behavior of the continue in the following function call
6615          # results in unspecified behavior (if execution reaches this
6616          # point):
6617          do_continue
6618
6619          trap 'continue 2' USR1
6620          echo " sending SIGUSR1 to self ($j)...""
6621          # the behavior of the continue in the trap invoked from the
6622          # following signal results in unspecified behavior (if
6623          # execution reaches this point):
6624          kill -USR1 $$
6625          sleep 1
6626      done
6627  }
6628  for i in 1 2; do
6629      echo "running foo ($i)...""
6630      foo
6631  done

```

6632 On Page: 2339 Line: 73966 Section: 2.14 continue  
 6633 (2013 edition Page: 2362 Line: 75244)

6634 Change the DESCRIPTION section from:

6635 The *continue* utility shall return to the top of the smallest enclosing **for**, **while**, or **until** loop, or to the top  
 6636 of the *n*th enclosing loop, if *n* is specified. This involves repeating the condition list of a **while** or **until** loop  
 6637 or performing the next assignment of a **for** loop, and re-executing the loop if appropriate.

6638 The value of *n* is a decimal integer greater than or equal to 1. The default shall be equivalent to *n*=1. If *n* is  
 6639 greater than the number of enclosing loops, the outermost enclosing loop shall be used.

6640 to:

6641 If *n* is specified, the *continue* utility shall return to the top of the *n*th enclosing **for**, **while**, or **until** loop. If *n*  
 6642 is not specified, *continue* shall behave as if *n* was specified as 1. Returning to the top of the loop involves  
 6643 repeating the condition list of a **while** or **until** loop or performing the next assignment of a **for** loop, and re-  
 6644 executing the loop if appropriate. The value of *n* is a positive decimal integer. If *n* is greater than the  
 6645 number of enclosing loops, the outermost enclosing loop shall be used. If there is no enclosing loop, the  
 6646 behavior is unspecified.

6647 The meaning of "enclosing" shall be as specified in the description of the *break* utility.

6648 *Rationale:* Austin Group Defect Report(s) applied: 842. See <http://austingroupbugs.net/view.php?id=842>.

6649

6650 **Change Number: XCU/TC2/D4/0047** [717]

6651 On Page: 2347 Line: 74214 Section: 2.14 exit  
 6652 (2013 edition Page: 2370 Line: 75508)

6653 In the DESCRIPTION section, change from:

6654 The *exit* utility shall cause the shell to exit with the exit status specified by the unsigned decimal integer *n*.

6655 to:

6656 The *exit* utility shall cause the shell to exit from its current execution environment with the exit status  
 6657 specified by the unsigned decimal integer *n*. If the current execution environment is a subshell  
 6658 environment, the shell shall exit from the subshell environment with the specified exit status and continue  
 6659 in the environment from which that subshell environment was invoked, otherwise the shell utility shall  
 6660 terminate with the specified exit status.

6661 *Rationale*: Austin Group Defect Report(s) applied: 717. See <http://austingroupbugs.net/view.php?id=717>.  
 6662 The changes made here make the standard reflect existing practice.

6663 **Change Number: XCU/TC2/D4/0048** [960]

6664 On Page: 2347 Line: 74240 Section: 2.14 exit  
 6665 (2013 edition Page: 2370 Line: 75534)

6666 In the EXIT STATUS section, change from:

6667 The exit status shall be *n*, if specified.

6668 to:

6669 The exit status shall be *n*, if specified, except that the behavior is unspecified if *n* is not an unsigned  
 6670 decimal integer or is greater than 255.

6671 *Rationale*: Austin Group Defect Report(s) applied: 960. See <http://austingroupbugs.net/view.php?id=960>.

6672 **Change Number: XCU/TC2/D4/0049** [717]

6673 On Page: 2348 Line: 74252 Section: 2.14 exit  
 6674 (2013 edition Page: 2371 Line: 75546)

6675 In the EXAMPLES section, add at the end of the section:

6676 Propagate error handling from within a subshell:

```
6677 (
6678     command1 || exit 1
6679     command2 || exit 1
6680     exec command3
6681 ) > outputfile || exit 1
6682 echo "outputfile created successfully"
```

6683 *Rationale*: Austin Group Defect Report(s) applied: 717. See <http://austingroupbugs.net/view.php?id=717>.

6684 **Change Number: XCU/TC2/D4/0050** [960]

6685 On Page: 2348 Line: 74258 Section: 2.14 exit  
 6686 (2013 edition Page: 2371 Line: 75552)

6687 In the RATIONALE section, add a new paragraph:

6688 The behavior of *exit* when given an invalid argument or unknown option is unspecified, because of  
 6689 differing practices in the various historical implementations. A value larger than 255 might be truncated by  
 6690 the shell, and be unavailable even to a parent process that uses *waitid()* to get the full exit value. It is  
 6691 recommended that implementations that detect any usage error should cause a non-zero exit status (or, if  
 6692 the shell is interactive and the error does not cause the shell to abort, store a non-zero value in \$?), but even  
 6693 this was not done historically in all shells.

6694 *Rationale*: Austin Group Defect Report(s) applied: 960. See <http://austingroupbugs.net/view.php?id=960>.

6695 **Change Number: XCU/TC2/D4/0051** [654]

6696 On Page: 2349 Line: 74290 Section: 2.14 export  
 6697 (2013 edition Page: 2372 Line: 75584)

6698 No change required to POSIX.1-2008.

6699 Revert Change Number XCU/TC1/D5/0043 from TC1, removing the text added:

6700 If a variable assignment precedes the command name of *export* but that variable is not also listed as an  
 6701 operand of *export*, then that variable shall be set in the current shell execution environment after the  
 6702 completion of the *export* command, but it is unspecified whether that variable is marked for export.

6703 *Rationale*: Austin Group Defect Report(s) applied: 654. See <http://austingroupbugs.net/view.php?id=654>.  
 6704 This change is layered on top of Change Number: XCU/TC1/D5/0043 from TC1. Earlier changes in section  
 6705 2.9.1 now mean that the change in TC1 is not required.

6706 **Change Number: XCU/TC2/D4/0052** [960]

6707 On Page: 2350 Line: 74312 Section: 2.14 export  
 6708 (2013 edition Page: 2373 Line: 75609)

6709 In the EXIT STATUS section, change from:

6710 Zero.

6711 to:

0 All *name* operands were successfully exported.

>0 At least one *name* could not be exported, or the **-p** option was specified and an error occurred.

6712

6713

6714 On Page: 2350 Line: 74316 Section: 2.14 export  
 6715 (2013 edition Page: 2373 Line: 75613)

6716 In the APPLICATION USAGE section, change from:

6717 None.

6718 to:

6719 Note that, unless X was previously marked readonly, the value of \$? after

6720 `export X=$(false)`

6721 will be 0 (because *export* successfully set X to the empty string) and that execution continues even if *set -e*  
 6722 is in effect. In order to detect command substitution failures, a user must separate the assignment from the  
 6723 *export*, as in

6724 `X=$(false)`  
 6725 `export X`

6726 On Page: 2353 Line: 74396 Section: 2.14 readonly  
 6727 (2013 edition Page: 2376 Line: 75695)

6728 In the EXIT STATUS section, change from:

6729 Zero.

6730 to:

0 All *name* operands were successfully marked readonly.

>0 At least one *name* could not be marked readonly, or the **-p** option was specified and an error occurred.

6731 On Page: 2355 Line: 74460 Section: 2.14 return  
 6732 (2013 edition Page: 2378 Line: 25759)

6733 In the EXIT STATUS section, change from:

6734 If the value of *n* is greater than 255, the results are undefined.

6735 to:

6736 If *n* is not an unsigned decimal integer, or is greater than 255, the results are unspecified.

6737 *Rationale:* Austin Group Defect Report(s) applied: 960. See <http://austingroupbugs.net/view.php?id=960>.  
 6738 The standard was unclear whether certain shell special builtins must have exit status of 0, or whether the  
 6739 overall rule for non-zero status after error detection applied. Existing shells have non-zero status after  
 6740 errors.

6741

6742 **Change Number: XCU/TC2/D4/0053** [584]

6743 On Page: 2357 Line: 74508, 74510 Section: 2.14 set

6744 In the DESCRIPTION section, change from:

6745 <hyphen>

6746 to:

6747 <hyphen-minus>

6748 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

6749 **Change Number: XCU/TC2/D4/0054** [717]

6750 On Page: 2358 Line: 74545 Section: 2.14 set

6751 (2013 edition Page: 2381 Line: 75845)

6752 In the DESCRIPTION section, replace the description of **-e** with:

6753 When this option is on, when any command fails (for any of the reasons listed in Section 2.8.1 (on page  
 6754 xxx) or by returning an exit status greater than zero) the shell immediately shall exit, as if by executing the  
 6755 *exit* special built-in utility with no arguments, with the following exceptions:

- 6756 1. The failure of any individual command in a multi-command pipeline shall not cause the shell to  
 6757 exit. Only the failure of the pipeline itself shall be considered.
- 6758 2. The **-e** setting shall be ignored when executing the compound list following the **while**, **until**, **if**, or  
 6759 **elif** reserved word, a pipeline beginning with the **!** reserved word, or any command of an AND-OR  
 6760 list other than the last.
- 6761 3. If the exit status of a compound command other than a subshell command was the result of a  
 6762 failure while **-e** was being ignored, then **-e** shall not apply to this command.

6763 This requirement applies to the shell environment and each subshell environment separately. For example,  
 6764 in:

6765 `set -e; (false; echo one) | cat; echo two`

6766 the **false** command causes the subshell to exit without executing **echo one**; however, **echo two** is  
 6767 executed because the exit status of the pipeline **(false; echo one) | cat** is zero.

6768 *Rationale:* Austin Group Defect Report(s) applied: 717. See <http://austingroupbugs.net/view.php?id=717>.

6769 This is layered change on XCU/TC1/D5/0046 [52]. The change over the 2013 edition is to change from:  
 6770 the shell immediately shall exit with the following exceptions

6771 to:

6772 the shell immediately shall exit, as if by executing the **exit** special built-in utility with no arguments, with  
 6773 the following exceptions

6774 **Change Number: XCU/TC2/D4/0055** [717]

6775 On Page: 2359 Line: 74590 Section: 2.14 set

6776 In the DESCRIPTION section, change the description of **-u** from:

6777 The shell shall write a message to standard error when it tries to expand a variable that is not set and  
 6778 immediately exit. An interactive shell shall not exit.

6779 to:

6780 When the shell tries to expand an unset parameter other than the '@' and '\*' special parameters, it shall write  
 6781 a message to standard error and the expansion shall fail with the consequences specified in 2.8.1  
 6782 Consequences of Shell Errors (on page xxx).

6783 *Rationale:* Austin Group Defect Report(s) applied: 717. See <http://austingroupbugs.net/view.php?id=717>.  
 6784 This is a layered change on XCU/TC1/D5/0047 [155,280]. This change replaces the previous revised text  
 6785 for the -u option.

6786 **Change Number: XCU/TC2/D4/0056** [960]

6787 On Page: 2360 Line: 74627 Section: 2.14 set  
 6788 (2013 edition Page: 2383 Line: 75944)

6789 In the EXIT STATUS section, change from:

6790 Zero.

6791 to:

0 Successful completion.

>0 An invalid option was specified, or an error occurred.

6792 On Page: 2366 Line: 74829 Section: 2.14 times  
 6793 (2013 edition Page: 2389 Line: 76171)

6794 In the EXIT STATUS section, change from:

6795 Zero.

6796 to:

0 Successful completion.

>0 An error occurred.

6797 *Rationale:* Austin Group Defect Report(s) applied: 960. See <http://austingroupbugs.net/view.php?id=960>.  
 6798 The standard was unclear whether certain shell special builtins must have exit status of 0, or whether the  
 6799 overall rule for non-zero status after error detection applied. Existing shells have non-zero status after

6800 errors.

6801 **Change Number: XCU/TC2/D4/0057** [584]

6802 On Page: 2415 Line: 76517, 76518 Section: ar

6803 In the RATIONALE section, change from:

6804 <hyphen>

6805 to:

6806 <hyphen-minus>

6807 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

6808 **Change Number: XCU/TC2/D4/0058** [584]

6809 On Page: 2434 Line: 77280 Section: awk

6810 In the EXTENDED DESCRIPTION section, change from:

6811 sequence begins with a minus-sign,

6812 to:

6813 sequence begins with a <hyphen-minus>,

6814 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

6815 **Change Number: XCU/TC2/D4/0059** [963]

6816 On Page: 2437 Line: 77382 Section: awk  
(2013 edition Page: 2459 Line: 78745)

6818 In the EXTENDED DESCRIPTION section, change from:

6819 operands shall be converted to strings as required and a string comparison shall be made using the locale-specific collation sequence. The value of the comparison expression shall be 1 if the relation is true, or 0 if the relation is false.

6822 to:

6823 operands shall be converted to strings as required and a string comparison shall be made as follows:

6824 • For the "!=" and "==" operators, the strings should be compared to check if they are identical but  
6825 may be compared using the locale-specific collation sequence to check if they collate equally.  
6826 • For the other operators, the strings shall be compared using the locale-specific collation sequence.

6827 The value of the comparison expression shall be 1 if the relation is true, or 0 if the relation is false.

6828 *Rationale*: Austin Group Defect Report(s) applied: 963. See <http://austingroupbugs.net/view.php?id=963>.

6829 **Change Number: XCU/TC2/D4/0060** [226]

6830 On Page: 2447 Line: 77855 Section: awk  
 6831 (2013 edition Page: 2470 Line: 79219)

6832 In the EXTENDED DESCRIPTION section, apply the following *diff* to the grammar:

```

6833   program      : item_list
6834   -           | actionless_item_list
6835   +           | item_list item
6836   ;
6837
6838
6839   -item_list    : newline_opt
6840   -           | actionless_item_list item terminator
6841   -           | item_list      item terminator
6842   -           | item_list      action terminator
6843   +item_list    : /* empty */
6844   +           | item_list item terminator
6845   ;
6846
6847
6848   -actionless_item_list : item_list      pattern terminator
6849   -           | actionless_item_list pattern terminator
6850   -           ;
6851   -
6852
6853   -item      : pattern action
6854   +item      : action
6855   +           | pattern action
6856   +           | normal_pattern
6857   +           | Function NAME '(' param_list_opt ')'
6858   +           | newline_opt action
6859   +           | Function FUNC_NAME '(' param_list_opt ')'

```

6860 On Page: 2448 Line: 77878 Section: awk  
 6861 (2013 edition Page: 2471 Line: 79242)

6862 In the EXTENDED DESCRIPTION section, apply the following *diff* to the grammar:

```

6863   -pattern      : Begin
6864   -           | End
6865   -           | expr
6866   +pattern      : normal_pattern
6867   +           | special_pattern
6868   +           ;
6869
6870   +normal_pattern : expr
6871   +           | expr ',' newline_opt expr
6872   +           ;
6873
6874
6875   +special_pattern : Begin
6876   +           | End
6877   +           ;
6878
6879   +

```

```

6880    action      : '{' newline_opt      '}''
6881          | '{' newline_opt terminated_statement_list '}''
6882          | '{' newline_opt unterminated_statement_list '}''
6883      ;
6884
6885
6886    -terminator : terminator ';'      ''
6887    -          | terminator NEWLINE
6888    +terminator : terminator NEWLINE
6889          |      ';'      ''
6890          |      NEWLINE
6891      ;

```

6892 *Rationale*: Austin Group Defect Report(s) applied: 226. See <http://austingroupbugs.net/view.php?id=226>.  
 6893 The standard does not require support for awk '{print}; {print}', but this was unintentional.

6894 awk BEGIN (with no action) and awk END (with no action) are allowed by the grammar but forbidden by  
 6895 the EXTENDED DESCRIPTION section on Special Patterns (see Issue 7, P2440, L77545 and in Issue 7,  
 6896 2013 Edition, P2463, L78909).

6897 The standard requires support for awk '{print};;{print}', but this is not historic practice and is considered to  
 6898 be a poor programming practice.

6899 **Change Number: XCU/TC2/D4/0061** [663]

6900 On Page: 2455 Line: 78223 Section: awk  
 6901 (2013 edition Page: 2478 Line: 79587)

6902 In the APPLICATION USAGE section, add a new paragraph:

6903 When using *awk* to process pathnames, it is recommended that LC\_ALL, or at least LC\_CTYPE and  
 6904 LC\_COLLATE, are set to POSIX or C in the environment, since pathnames can contain byte sequences  
 6905 that do not form valid characters in some locales, in which case the utility's behavior would be undefined.  
 6906 In the POSIX locale each byte is a valid single-byte character, and therefore this problem is avoided.

6907 *Rationale*: Austin Group Defect Report(s) applied: 663. See <http://austingroupbugs.net/view.php?id=663>.

6908 **Change Number: XCU/TC2/D4/0062** [963]

6909 On Page: 2455 Line: 78223 Section: awk  
 6910 (2013 edition Page: 2478 Line: 79587)

6911 In the APPLICATION USAGE section, add two new paragraphs:

6912 On implementations where the "==" operator checks if strings collate equally, applications needing to  
 6913 check whether strings are identical can use:

6914 `length(a) == length(b) && index(a,b) == 1`

6915 On implementations where the "==" operator checks if strings are identical, applications needing to check  
 6916 whether strings collate equally can use:

6917 `a <= b && a >= b`

6918 *Rationale:* Austin Group Defect Report(s) applied: 963. See <http://austingroupbugs.net/view.php?id=963>.

6919 **Change Number:** XCU/TC2/D4/0063 [226]

6920 On Page: 2458 Line: 78313 Section: awk  
 6921 (2013 edition Page: 2481 Line: 79677)

6922 In the RATIONALE section, add a new paragraph:

6923 Earlier versions of this standard required implementations to support multiple adjacent <semicolon>s, lines  
 6924 with one or more <semicolon>s before a rule ("pattern {action}" pairs), and lines with only  
 6925 <semicolon>(s). These are not required by this standard and are considered poor programming practice, but  
 6926 can be accepted by an implementation of awk as an extension.

6927 *Rationale:* Austin Group Defect Report(s) applied: 226. See <http://austingroupbugs.net/view.php?id=226>.

6928 **Change Number:** XCU/TC2/D4/0064 [963]

6929 On Page: 2463 Line: 78550 Section: awk  
 6930 (2013 edition Page: 2486 Line: 79914)

6931 In the FUTURE DIRECTIONS section, change from:

6932 None.

6933 to:

6934 A future version of this standard may require the "!=" and "==" operators to perform string comparisons by  
 6935 checking if the strings are identical (and not by checking if they collate equally).

6936 *Rationale:* Austin Group Defect Report(s) applied: 963. See <http://austingroupbugs.net/view.php?id=963>.

6937 **Change Number:** XCU/TC2/D4/0065 [612]

6938 On Page: 2465 Line: 78660 Section: basename

6939 Add a new paragraph at the end of the EXAMPLES section :

6940 The EXAMPLES section of the *basename()* function (see XREF to XSH *basename()* EXAMPLES section)  
 6941 includes a table showing examples of the results of processing several sample pathnames by the  
 6942 *basename()* and *dirname()* functions and by the *basename* and *dirname* utilities.

6943 On Page: 2466 Line: 78676 Section: basename

6944 In the SEE ALSO section, add a new paragraph at the end:

6945 XSH *basename()*,

6946 *Rationale:* Austin Group Defect Report(s) applied: 612. See <http://austingroupbugs.net/view.php?id=612>.

6947

6948 **Change Number: XCU/TC2/D4/0066** [584]

6949 On Page: 2476 Line: 79064 Section: bc

6950 In the EXTENDED DESCRIPTION section, change from:

6951 <hyphen>

6952 to:

6953 <hyphen-minus>

6954 *Rationale*: Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

6955 **Change Number: XCU/TC2/D4/0067** [679]

6956 On Page: 2480 Line: 79259 Section: bc

6957 (2013 edition Page: 2503 Line: 80630)

6958 In the EXTENDED DESCRIPTION section, change from:

6959 **j( expression, expression )**

6960 Bessel function of integer order.

6961 to:

6962 **j( expression1, expression2 )**

6963 Bessel function of *expression2* of the first kind of integer order *expression1*.

6964 *Rationale*: Austin Group Defect Report(s) applied: 679. See <http://austingroupbugs.net/view.php?id=679>.

6965 **Change Number: XCU/TC2/D4/0068** [650]

6966 On Page: 2489 Line: 79568 Section: c99

6967 In the OPTIONS section, in the -E description change from:

6968 expanding all preprocessor directives

6969 to:

6970 executing all preprocessor directives

6971 *Rationale*: Austin Group Defect Report(s) applied: 650. See <http://austingroupbugs.net/view.php?id=650>.

6972 **Change Number: XCU/TC2/D4/0069** [670]

6973 On Page: 2491 Line: 79683,79696 Section: c99

6974 In the EXTENDED DESCRIPTION section, at line 79683 (-I c), change from:

6975 beginning with the prefix clock\_ or time\_

6976 to:

6977 beginning with the prefix clock\_ or timer\_

6978 At line 79696 (-l rt), change from:

6979 interfaces beginning with the prefix clock\_ and time\_

6980 to:

6981 interfaces beginning with the prefix clock\_ and timer\_

6982 *Rationale:* Austin Group Defect Report(s) applied: 670. See <http://austingroupbugs.net/view.php?id=670>.

6983 **Change Number:** XCU/TC2/D4/0070 [638]

6984 On Page: 2494 Line: 79799 Section: c99

6985 In the CONSEQUENCES OF ERRORS section, change from:

6986 it shall not perform the link phase and return a non-zero exit status

6987 to:

6988 it shall not perform the link phase and it shall return a non-zero exit status

6989 *Rationale:* Austin Group Defect Report(s) applied: 638. See <http://austingroupbugs.net/view.php?id=638>.

6990 **Change Number:** XCU/TC2/D4/0071 [650]

6991 On Page: 2497 Line: 79928 Section: c99

6992 In the RATIONALE section, add a new paragraph to the end of the section:

6993 When the -E option is used, execution of some #pragma preprocessing directives may simply result in a  
6994 copy of the directive being included in the output as part of the allowed extra information used by  
6995 subsequent compilation passes (see STDOUT).

6996 *Rationale:* Austin Group Defect Report(s) applied: 650. See <http://austingroupbugs.net/view.php?id=650>.

6997 **Change Number:** XCU/TC2/D4/0072 [784]

6998 On Page: 2497 Line: 79930 Section: c99

6999 (2013 edition Page: 2520 Line: 81307)

7000 In the FUTURE DIRECTIONS section, change from:

7001 None.

7002 to:

7003 Unlike all of the other non-OB-shaded utilities in this standard, a utility by this name probably will not  
7004 appear in the next revision of this standard. This utility's name is tied to the current revision of the C  
7005 Standard at the time this standard is approved. Since the C Standard and this standard are maintained by  
7006 different organizations on different schedules, we cannot predict what the compiler will be named in the  
7007 next revision of the standard.

7008 *Rationale:* Austin Group Defect Report(s) applied: 784. See <http://austingroupbugs.net/view.php?id=784>.

7009 **Change Number:** XCU/TC2/D4/0073 [876]

7010 On Page: 2502 Line: 80083 Section: cat  
7011 (2013 edition Page: 2526 Line: 81474)

7012 In the STDOUT section, add a new sentence:

7013 If the standard output is a regular file, and is the same file as any of the input file operands, the  
7014 implementation may treat this as an error.

7015 On Page: 2502 Line: 80115-80118 Section: cat  
7016 (2013 edition Page: 2526 Line: 81506-81509)

7017 In the EXAMPLES section, change from:

7018 Because of the shell language mechanism used to perform output redirection, a command such as this:

7019 `cat doc doc.end > doc`

7020 causes the original data in `doc` to be lost.

7021 to:

7022 Because of the shell language mechanism used to perform output redirection, a command such as this:

7023 `cat doc doc.end >> doc`

7024 causes the original data in `doc` to be lost before `cat` even begins execution. This is true whether the `cat`  
7025 command fails with an error or silently succeeds (the specification allows both behaviors). In order to  
7026 append the contents of `doc.end` without losing the original contents of `doc`, this command should be used  
7027 instead:

7028 `cat doc.end >> doc`

7029 *Rationale:* Austin Group Defect Report(s) applied: 876. See <http://austingroupbugs.net/view.php?id=876>.  
7030 A cat command which redirects its standard output to a file that is also named as a file operand is likely to  
7031 run until the output file reaches the maximum output file size allowed for that process or the underlying  
7032 filesystem runs out of space. This is a common application error that accidentally consumes a lot of space  
7033 needed by other users on the system. Therefore, many implementations of the cat utility check for this  
7034 condition and, when it is found, print a diagnostic message and exit with a non-zero exit status. This  
7035 behavior is not currently allowed by the standard, but should be.

7036 **Change Number: XCU/TC2/D4/0074** [584]

7037 On Page: 2507 Line: 80264 Section: cd

7038 In the OPERANDS section, change from:

7039 <hyphen>

7040 to:

7041 <hyphen-minus>

7042 *Rationale:* Austin Group Defect Report(s) applied: 584. See <http://austingroupbugs.net/view.php?id=584>.

7043 **Change Number: XCU/TC2/D4/0075** [478]

7044 On Page: 2532 Line: 81246 Section: cmp

7045 In the OPTIONS section, change the -s description from:

7046 Write nothing for differing files; return exit status only.

7047 to:

7048 Write nothing to standard output or standard error when files differ; indicate differing files through exit status only.

7049 It is unspecified whether a diagnostic message is written to standard error when an error is encountered; if a message is not written, the error is indicated through exit status only.

7050

7051 On Page: 2533 Line: 81295 Section: cmp

7052 In the STDERR section, add a new paragraph to the end of the section:

7053 If the -s option is used and an error occurs, it is unspecified whether a diagnostic message is written to standard error.

7054

7055 On Page: 2534 Line: 81314 Section: cmp

7056 In the APPLICATION USAGE section, add two new paragraphs to the end of the section:

7057 Since the behavior of -s differs between implementations as to whether error messages are written, the only

7058 way to ensure consistent behavior of cmp when -s is used is to redirect standard error to /dev/null.

7059 If error messages are wanted, instead of using -s standard output should be redirected to /dev/null, and

7060 anything written to standard error should be discarded if the exit status is 1. For example:

```

7061     silent_cmp() {
7062         # compare files with no output except error messages
7063         message=$(cmp "$@" 2>&1 >/dev/null)
7064         status=$?
7065         case $status in
7066             (0|1) ;;
7067             (*) printf '%s\n' "$message" ;;

```