

1 / *Technical Standard*

2 **System Interfaces, Issue 6**

3 *The Open Group*  
4 *The Institute of Electrical and Electronics Engineers, Inc.*

5



6 © 2000, *The Open Group*

7 © 2000, *The Institute of Electrical and Electronics Engineers, Inc.*

8 All rights reserved.

9 Except as permitted below, no part of this publication may be reproduced, stored in a retrieval  
10 system, or transmitted, in any form or by any means, electronic, mechanical, photocopying,  
11 recording or otherwise, without the prior permission of the copyright owners.

This is an unapproved draft, subject to change. Permission is hereby granted for Austin Group participants to reproduce IEEE Std. 1003.1-200x for purposes of IEEE, The Open Group, and JTC1 standardization activities.

Other entities seeking permission to reproduce IEEE Std. 1003.1-200x for standardization purposes or activities must contact the copyright owners for an appropriate license. Use of information contained within this unapproved draft is at your own risk.

Portions of IEEE Std. 1003.1-200x are derived with permission from copyrighted material owned by Hewlett-Packard Company, International Business Machines Corporation, Novell Inc., The Open Software Foundation, and Sun Microsystems, Inc.

12 Technical Standard

13 System Interfaces, Issue 6

14 Document Number:

15 Published in the U.K. by The Open Group, 2000.

16 See <http://www.opengroup.org/austin/bugreport.html> for instructions on commenting on this  
17 unapproved draft document.

# Contents

19	<b>Chapter 1</b>	<b>Introduction.....</b>	<b>491</b>
20	1.1	Scope.....	491
21	1.2	Conformance .....	491
22	1.3	Normative References .....	491
23	1.4	Changes from Issue 4.....	491
24	1.4.1	Changes from Issue 4 to Issue 4, Version 2.....	491
25	1.4.2	Changes from Issue 4, Version 2 to Issue 5.....	492
26	1.4.3	Changes from Issue 5 to Issue 6 (IEEEStd.1003.1-200x) .....	492
27	1.5	New Features.....	493
28	1.5.1	New Features in Issue 4, Version 2.....	493
29	1.5.2	New Features in Issue 5.....	494
30	1.5.3	New Features in Issue 6.....	496
31	1.6	Terminology .....	497
32	1.7	Definitions.....	499
33	1.8	Relationship to Other Formal Standards .....	500
34	1.9	Portability .....	501
35	1.9.1	Codes.....	501
36	1.10	Format of Entries.....	509
37	<b>Chapter 2</b>	<b>General Information .....</b>	<b>511</b>
38	2.1	Use and Implementation of Functions.....	511
39	2.2	The Compilation Environment.....	512
40	2.2.1	POSIX.1 Symbols .....	512
41	2.2.1.1	The <code>_POSIX_C_SOURCE</code> Feature Test Macro.....	512
42	2.2.1.2	The <code>_XOPEN_SOURCE</code> Feature Test Macro.....	512
43	2.2.2	The Name Space.....	513
44	2.3	Error Numbers.....	521
45	2.3.1	Additional Error Numbers.....	527
46	2.4	Signal Concepts.....	528
47	2.4.1	Signal Generation and Delivery.....	528
48	2.4.2	Realtime Signal Generation and Delivery .....	529
49	2.4.3	Signal Actions.....	530
50	2.4.4	Signal Effects on Other Functions .....	534
51	2.5	Standard I/O Streams.....	535
52	2.5.1	Interaction of File Descriptors and Standard I/O Streams.....	535
53	2.5.2	Stream Orientation and Encoding Rules .....	537
54	2.6	STREAMS.....	539
55	2.6.1	Accessing STREAMS.....	540
56	2.7	XSI Interprocess Communication .....	541
57	2.7.1	IPC General Description.....	541
58	2.8	Realtime .....	543
59	2.8.1	Realtime Signals.....	543

60	2.8.2	Asynchronous I/O .....	543
61	2.8.3	Memory Management .....	545
62	2.8.3.1	Memory Locking.....	545
63	2.8.3.2	Memory Mapped Files.....	545
64	2.8.3.3	Memory Protection.....	545
65	2.8.3.4	Typed Memory Objects .....	546
66	2.8.4	Process Scheduling.....	546
67	2.8.5	Clocks and Timers .....	550
68	2.9	Threads.....	553
69	2.9.1	Thread-Safety.....	553
70	2.9.2	Thread IDs.....	554
71	2.9.3	Thread Mutexes.....	554
72	2.9.4	Thread Scheduling.....	555
73	2.9.5	Thread Cancellation .....	557
74	2.9.5.1	Cancelability States .....	557
75	2.9.5.2	Cancellation Points .....	558
76	2.9.5.3	Thread Cancellation Cleanup Handlers.....	560
77	2.9.5.4	Async-Cancel Safety.....	560
78	2.9.6	Thread Read-Write Locks.....	560
79	2.9.7	Thread Interactions with Regular File Operations .....	561
80	2.10	Sockets.....	562
81	2.10.1	Protocol Families.....	562
82	2.10.2	Protocols .....	562
83	2.10.3	Addressing .....	562
84	2.10.4	Routing.....	562
85	2.10.5	Interfaces.....	563
86	2.10.6	Socket Types.....	563
87	2.10.7	Socket I/O Mode.....	563
88	2.10.8	Socket Owner.....	564
89	2.10.9	Socket Queue Limits .....	564
90	2.10.10	Pending Error.....	564
91	2.10.11	Socket Receive Queue.....	564
92	2.10.12	Socket Out-of-Band Data State .....	565
93	2.10.13	Connection Indication Queue .....	565
94	2.10.14	Signals .....	565
95	2.10.15	Asynchronous Errors .....	565
96	2.10.16	Use of Options.....	566
97	2.10.17	Use of Sockets for Local UNIX Connections.....	569
98	2.10.17.1	Headers .....	569
99	2.10.18	Use of Sockets over Internet Protocols Based on IPv4.....	569
100	2.10.18.1	Headers .....	569
101	2.10.19	Use of Sockets over Internet Protocols Based on IPv6.....	570
102	2.10.19.1	Addressing .....	570
103	2.10.19.2	Compatibility with IPv4.....	571
104	2.10.19.3	Interface Identification.....	571
105	2.10.19.4	Options.....	571
106	2.10.19.5	Headers .....	572
107	2.11	Tracing.....	573

108	2.11.1	Tracing Data Definitions .....	574
109	2.11.1.1	Structures.....	574
110	2.11.1.2	Trace Stream Attributes.....	578
111	2.11.2	Trace Event Type Definitions .....	579
112	2.11.2.1	System Trace Event Type Definitions.....	579
113	2.11.2.2	User Trace Event Type Definitions.....	581
114	2.11.3	Trace Functions.....	582
115	2.12	Data Types.....	583
116	<b>Chapter 3</b>	<b>System Interfaces .....</b>	<b>585</b>
117		<i>FD_CLR()</i> .....	586
118		<i>_Exit()</i> .....	587
119		<i>_longjmp()</i> .....	588
120		<i>_setjmp()</i> .....	590
121		<i>_tolower()</i> .....	591
122		<i>_toupper()</i> .....	592
123		<i>a64l()</i> .....	593
124		<i>abort()</i> .....	595
125		<i>abs()</i> .....	597
126		<i>accept()</i> .....	598
127		<i>access()</i> .....	600
128		<i>acos()</i> .....	603
129		<i>acosh()</i> .....	605
130		<i>aio_cancel()</i> .....	607
131		<i>aio_error()</i> .....	609
132		<i>aio_fsync()</i> .....	610
133		<i>aio_read()</i> .....	612
134		<i>aio_return()</i> .....	615
135		<i>aio_suspend()</i> .....	616
136		<i>aio_write()</i> .....	618
137		<i>alarm()</i> .....	621
138		<i>asctime()</i> .....	623
139		<i>asin()</i> .....	626
140		<i>asinh()</i> .....	628
141		<i>assert()</i> .....	629
142		<i>atan()</i> .....	630
143		<i>atan2()</i> .....	632
144		<i>atanh()</i> .....	634
145		<i>atexit()</i> .....	635
146		<i>atof()</i> .....	636
147		<i>atoi()</i> .....	637
148		<i>atol()</i> .....	639
149		<i>basename()</i> .....	641
150		<i>bcmp()</i> .....	643
151		<i>bcopy()</i> .....	644
152		<i>bind()</i> .....	645
153		<i>bsd_signal()</i> .....	647
154		<i>bsearch()</i> .....	649

155	<i>btowc()</i> .....	652
156	<i>bzero()</i> .....	653
157	<i>cabs()</i> .....	654
158	<i>cacos()</i> .....	655
159	<i>cacosh()</i> .....	656
160	<i>calloc()</i> .....	657
161	<i>carg()</i> .....	659
162	<i>casin()</i> .....	660
163	<i>casinh()</i> .....	661
164	<i>catan()</i> .....	662
165	<i>catanh()</i> .....	663
166	<i>catclose()</i> .....	664
167	<i>catgets()</i> .....	665
168	<i>catopen()</i> .....	667
169	<i>cbrt()</i> .....	669
170	<i>ccos()</i> .....	670
171	<i>ccosh()</i> .....	671
172	<i>ceil()</i> .....	672
173	<i>cexp()</i> .....	674
174	<i>cfgetispeed()</i> .....	675
175	<i>cfgetospeed()</i> .....	677
176	<i>cfsetispeed()</i> .....	678
177	<i>cfsetospeed()</i> .....	680
178	<i>chdir()</i> .....	682
179	<i>chmod()</i> .....	684
180	<i>chown()</i> .....	688
181	<i>cimag()</i> .....	692
182	<i>clearerr()</i> .....	693
183	<i>clock()</i> .....	694
184	<i>clock_getcpuclockid()</i> .....	696
185	<i>clock_getres()</i> .....	697
186	<i>clock_nanosleep()</i> .....	700
187	<i>clog()</i> .....	703
188	<i>close()</i> .....	704
189	<i>closedir()</i> .....	708
190	<i>closelog()</i> .....	710
191	<i>confstr()</i> .....	714
192	<i>conj()</i> .....	717
193	<i>connect()</i> .....	718
194	<i>copysign()</i> .....	721
195	<i>cos()</i> .....	722
196	<i>cosh()</i> .....	724
197	<i>cpow()</i> .....	726
198	<i>cproj()</i> .....	727
199	<i>creal()</i> .....	728
200	<i>creat()</i> .....	729
201	<i>crypt()</i> .....	731
202	<i>csin()</i> .....	733

# Contents

203	<i>csinh()</i> .....	734
204	<i>csqrt()</i> .....	735
205	<i>ctan()</i> .....	736
206	<i>ctanh()</i> .....	737
207	<i>ctermid()</i> .....	738
208	<i>ctime()</i> .....	740
209	<i>daylight</i> .....	742
210	<i>dbm_clearerr()</i> .....	743
211	<i>difftime()</i> .....	746
212	<i>dirname()</i> .....	747
213	<i>div()</i> .....	749
214	<i>dlclose()</i> .....	750
215	<i>dLError()</i> .....	752
216	<i>dlopen()</i> .....	754
217	<i>dlsym()</i> .....	757
218	<i>drand48()</i> .....	759
219	<i>dup()</i> .....	762
220	<i>ecvt()</i> .....	765
221	<i>encrypt()</i> .....	767
222	<i>endgrent()</i> .....	769
223	<i>endhostent()</i> .....	771
224	<i>endnetent()</i> .....	773
225	<i>endprotoent()</i> .....	775
226	<i>endpwent()</i> .....	777
227	<i>endservent()</i> .....	779
228	<i>endutxent()</i> .....	781
229	<i>environ</i> .....	784
230	<i>erand48()</i> .....	785
231	<i>erf()</i> .....	786
232	<i>errno</i> .....	788
233	<i>exec</i> .....	790
234	<i>exit()</i> .....	801
235	<i>exp()</i> .....	806
236	<i>exp2()</i> .....	808
237	<i>expm1()</i> .....	810
238	<i>fabs()</i> .....	811
239	<i>fattach()</i> .....	813
240	<i>fchdir()</i> .....	816
241	<i>fchmod()</i> .....	817
242	<i>fchown()</i> .....	819
243	<i>fclose()</i> .....	821
244	<i>fcntl()</i> .....	823
245	<i>fcvt()</i> .....	830
246	<i>fdatasync()</i> .....	831
247	<i>fdetach()</i> .....	832
248	<i>fdim()</i> .....	834
249	<i>fdopen()</i> .....	835
250	<i>feclearexcept()</i> .....	838

251	<i>fegetenv()</i> .....	839
252	<i>fegetexceptflag()</i> .....	840
253	<i>fegetround()</i> .....	841
254	<i>feholdexcept()</i> .....	843
255	<i>feof()</i> .....	844
256	<i>feraiseexcept()</i> .....	845
257	<i>ferror()</i> .....	846
258	<i>fesetenv()</i> .....	847
259	<i>fesetexceptflag()</i> .....	848
260	<i>fesetround()</i> .....	849
261	<i>fetestexcept()</i> .....	850
262	<i>feupdateenv()</i> .....	852
263	<i>fflush()</i> .....	853
264	<i>ffs()</i> .....	856
265	<i>fgetc()</i> .....	857
266	<i>fgetpos()</i> .....	860
267	<i>fgets()</i> .....	862
268	<i>fgetwc()</i> .....	864
269	<i>fgetws()</i> .....	866
270	<i>fileno()</i> .....	868
271	<i>flockfile()</i> .....	869
272	<i>floor()</i> .....	871
273	<i>fma()</i> .....	873
274	<i>fmax()</i> .....	874
275	<i>fmin()</i> .....	875
276	<i>fmod()</i> .....	876
277	<i>fmtmsg()</i> .....	878
278	<i>fnmatch()</i> .....	881
279	<i>fopen()</i> .....	883
280	<i>fork()</i> .....	887
281	<i>fpathconf()</i> .....	892
282	<i>fpclassify()</i> .....	897
283	<i>fprintf()</i> .....	898
284	<i>fputc()</i> .....	910
285	<i>fputs()</i> .....	912
286	<i>fputwc()</i> .....	914
287	<i>fputws()</i> .....	916
288	<i>fread()</i> .....	917
289	<i>free()</i> .....	919
290	<i>freeaddrinfo()</i> .....	921
291	<i>freehostent()</i> .....	925
292	<i>freopen()</i> .....	926
293	<i>frexp()</i> .....	930
294	<i>fscanf()</i> .....	932
295	<i>fseek()</i> .....	939
296	<i>fsetpos()</i> .....	942
297	<i>fstat()</i> .....	944
298	<i>fstatvfs()</i> .....	947



## Contents

299	<i>fsync()</i> .....	950
300	<i>ftell()</i> .....	952
301	<i>ftime()</i> .....	954
302	<i>ftok()</i> .....	956
303	<i>ftruncate()</i> .....	958
304	<i>ftrylockfile()</i> .....	960
305	<i>ftw()</i> .....	961
306	<i>funlockfile()</i> .....	964
307	<i>fwide()</i> .....	965
308	<i>fwprintf()</i> .....	966
309	<i>fwrite()</i> .....	973
310	<i>fwscanf()</i> .....	975
311	<i>gai_strerror()</i> .....	981
312	<i>gcvt()</i> .....	982
313	<i>getaddrinfo()</i> .....	983
314	<i>getc()</i> .....	984
315	<i>getc_unlocked()</i> .....	985
316	<i>getchar()</i> .....	987
317	<i>getcontext()</i> .....	988
318	<i>getcwd()</i> .....	990
319	<i>getdate()</i> .....	993
320	<i>getegid()</i> .....	998
321	<i>getenv()</i> .....	999
322	<i>geteuid()</i> .....	1002
323	<i>getgid()</i> .....	1003
324	<i>getgrent()</i> .....	1004
325	<i>getgrgid()</i> .....	1005
326	<i>getgrnam()</i> .....	1008
327	<i>getgroups()</i> .....	1011
328	<i>gethostbyaddr()</i> .....	1013
329	<i>gethostbyname()</i> .....	1017
330	<i>gethostent()</i> .....	1018
331	<i>gethostid()</i> .....	1019
332	<i>gethostname()</i> .....	1020
333	<i>getipnodebyaddr()</i> .....	1021
334	<i>getipnodebyname()</i> .....	1022
335	<i>getitimer()</i> .....	1023
336	<i>getlogin()</i> .....	1025
337	<i>getmsg()</i> .....	1028
338	<i>getnameinfo()</i> .....	1032
339	<i>getnetbyaddr()</i> .....	1034
340	<i>getnetbyname()</i> .....	1035
341	<i>getnetent()</i> .....	1036
342	<i>getopt()</i> .....	1037
343	<i>getpeername()</i> .....	1042
344	<i>getpgid()</i> .....	1043
345	<i>getpgrp()</i> .....	1044
346	<i>getpid()</i> .....	1045

347	<i>getpmsg()</i> .....	1046
348	<i>getppid()</i> .....	1047
349	<i>getpriority()</i> .....	1048
350	<i>getprotobyname()</i> .....	1051
351	<i>getprotobynumber()</i> .....	1052
352	<i>getprotoent()</i> .....	1053
353	<i>getpwent()</i> .....	1054
354	<i>getpwnam()</i> .....	1055
355	<i>getpwuid()</i> .....	1058
356	<i>getrlimit()</i> .....	1061
357	<i>getrusage()</i> .....	1064
358	<i>gets()</i> .....	1066
359	<i>getservbyname()</i> .....	1068
360	<i>getservbyport()</i> .....	1069
361	<i>getservent()</i> .....	1070
362	<i>getsid()</i> .....	1071
363	<i>getsockname()</i> .....	1072
364	<i>getsockopt()</i> .....	1073
365	<i>getsubopt()</i> .....	1076
366	<i>gettimeofday()</i> .....	1079
367	<i>getuid()</i> .....	1080
368	<i>getutxent()</i> .....	1082
369	<i>getwc()</i> .....	1083
370	<i>getwchar()</i> .....	1084
371	<i>getwd()</i> .....	1085
372	<i>glob()</i> .....	1086
373	<i>gmtime()</i> .....	1090
374	<i>grantpt()</i> .....	1092
375	<i>h_errno</i> .....	1093
376	<i>hcreate()</i> .....	1094
377	<i>htonl()</i> .....	1097
378	<i>htons()</i> .....	1098
379	<i>hypot()</i> .....	1099
380	<i>iconv()</i> .....	1101
381	<i>iconv_close()</i> .....	1103
382	<i>iconv_open()</i> .....	1104
383	<i>if_freenameindex()</i> .....	1106
384	<i>if_indextoname()</i> .....	1107
385	<i>if_nameindex()</i> .....	1108
386	<i>if_nametoindex()</i> .....	1109
387	<i>ilogb()</i> .....	1110
388	<i>imaxabs()</i> .....	1111
389	<i>imaxdiv()</i> .....	1112
390	<i>index()</i> .....	1113
391	<i>inet_addr()</i> .....	1114
392	<i>inet_lnaof()</i> .....	1116
393	<i>inet_makeaddr()</i> .....	1117
394	<i>inet_netof()</i> .....	1118

## Contents

395	<i>inet_network()</i> .....	1119
396	<i>inet_ntoa()</i> .....	1120
397	<i>inet_ntop()</i> .....	1121
398	<i>initstate()</i> .....	1123
399	<i>insque()</i> .....	1125
400	<i>ioctl()</i> .....	1128
401	<i>isalnum()</i> .....	1140
402	<i>isalpha()</i> .....	1141
403	<i>isascii()</i> .....	1142
404	<i>isastream()</i> .....	1143
405	<i>isatty()</i> .....	1144
406	<i>isblank()</i> .....	1145
407	<i>iscntrl()</i> .....	1146
408	<i>isdigit()</i> .....	1147
409	<i>isfdtype()</i> .....	1148
410	<i>isfinite()</i> .....	1149
411	<i>isgraph()</i> .....	1150
412	<i>isgreater()</i> .....	1151
413	<i>isgreaterequal()</i> .....	1152
414	<i>isinf()</i> .....	1153
415	<i>isless()</i> .....	1154
416	<i>islessequal()</i> .....	1155
417	<i>islessgreater()</i> .....	1156
418	<i>islower()</i> .....	1157
419	<i>isnan()</i> .....	1159
420	<i>isnormal()</i> .....	1160
421	<i>isprint()</i> .....	1161
422	<i>ispunct()</i> .....	1162
423	<i>isspace()</i> .....	1163
424	<i>isunordered()</i> .....	1164
425	<i>isupper()</i> .....	1165
426	<i>iswalnum()</i> .....	1166
427	<i>iswalpha()</i> .....	1168
428	<i>iswblank()</i> .....	1170
429	<i>iswcntrl()</i> .....	1171
430	<i>iswctype()</i> .....	1173
431	<i>iswdigit()</i> .....	1175
432	<i>iswgraph()</i> .....	1176
433	<i>iswlower()</i> .....	1178
434	<i>iswprint()</i> .....	1180
435	<i>iswpunct()</i> .....	1182
436	<i>iswspace()</i> .....	1184
437	<i>iswupper()</i> .....	1186
438	<i>iswxdigit()</i> .....	1188
439	<i>isxdigit()</i> .....	1189
440	<i>j0()</i> .....	1190
441	<i>rand48()</i> .....	1192
442	<i>kill()</i> .....	1193

443	<i>killpg()</i> .....	1196
444	<i>l64a()</i> .....	1197
445	<i>labs()</i> .....	1198
446	<i>lchown()</i> .....	1199
447	<i>lcong48()</i> .....	1201
448	<i>ldexp()</i> .....	1202
449	<i>ldiv()</i> .....	1204
450	<i>lfind()</i> .....	1205
451	<i>lgamma()</i> .....	1206
452	<i>link()</i> .....	1208
453	<i>lio_listio()</i> .....	1212
454	<i>listen()</i> .....	1215
455	<i>llrint()</i> .....	1217
456	<i>llround()</i> .....	1218
457	<i>localeconv()</i> .....	1219
458	<i>localtime()</i> .....	1223
459	<i>lockf()</i> .....	1226
460	<i>log()</i> .....	1229
461	<i>log10()</i> .....	1231
462	<i>log1p()</i> .....	1233
463	<i>log2()</i> .....	1234
464	<i>logb()</i> .....	1235
465	<i>longjmp()</i> .....	1236
466	<i>lrand48()</i> .....	1238
467	<i>lsearch()</i> .....	1239
468	<i>lseek()</i> .....	1241
469	<i>lstat()</i> .....	1243
470	<i>makecontext()</i> .....	1245
471	<i>malloc()</i> .....	1247
472	<i>mblen()</i> .....	1249
473	<i>mbrlen()</i> .....	1251
474	<i>mbrtowc()</i> .....	1253
475	<i>mbsinit()</i> .....	1255
476	<i>mbsrtowcs()</i> .....	1256
477	<i>mbstowcs()</i> .....	1258
478	<i>mbtowc()</i> .....	1260
479	<i>memccpy()</i> .....	1262
480	<i>memchr()</i> .....	1263
481	<i>memcmp()</i> .....	1264
482	<i>memcpy()</i> .....	1265
483	<i>memmove()</i> .....	1266
484	<i>memset()</i> .....	1267
485	<i>mkdir()</i> .....	1268
486	<i>mkfifo()</i> .....	1271
487	<i>mknod()</i> .....	1274
488	<i>mkstemp()</i> .....	1277
489	<i>mktemp()</i> .....	1279
490	<i>mktime()</i> .....	1281

## Contents

491	<i>mlock()</i> .....	1283
492	<i>mlockall()</i> .....	1285
493	<i>mmap()</i> .....	1287
494	<i>modf()</i> .....	1294
495	<i>mprotect()</i> .....	1296
496	<i>mq_close()</i> .....	1298
497	<i>mq_getattr()</i> .....	1299
498	<i>mq_notify()</i> .....	1301
499	<i>mq_open()</i> .....	1303
500	<i>mq_receive()</i> .....	1306
501	<i>mq_send()</i> .....	1309
502	<i>mq_setattr()</i> .....	1311
503	<i>mq_timedreceive()</i> .....	1313
504	<i>mq_timedsend()</i> .....	1314
505	<i>mq_unlink()</i> .....	1315
506	<i>rand48()</i> .....	1317
507	<i>msgctl()</i> .....	1318
508	<i>msgget()</i> .....	1320
509	<i>msgrcv()</i> .....	1322
510	<i>msgsnd()</i> .....	1325
511	<i>msync()</i> .....	1328
512	<i>munlock()</i> .....	1331
513	<i>munlockall()</i> .....	1332
514	<i>munmap()</i> .....	1333
515	<i>nan()</i> .....	1335
516	<i>nanosleep()</i> .....	1336
517	<i>nearbyint()</i> .....	1338
518	<i>nextafter()</i> .....	1339
519	<i>nftw()</i> .....	1341
520	<i>nice()</i> .....	1344
521	<i>nl_langinfo()</i> .....	1346
522	<i>nrnd48()</i> .....	1348
523	<i>ntohl()</i> .....	1349
524	<i>ntohs()</i> .....	1350
525	<i>open()</i> .....	1351
526	<i>opendir()</i> .....	1359
527	<i>openlog()</i> .....	1362
528	<i>optarg</i> .....	1363
529	<i>pathconf()</i> .....	1364
530	<i>pause()</i> .....	1365
531	<i>pclose()</i> .....	1367
532	<i>perror()</i> .....	1369
533	<i>pipe()</i> .....	1371
534	<i>poll()</i> .....	1373
535	<i>popen()</i> .....	1377
536	<i>posix_fadvise()</i> .....	1380
537	<i>posix_fallocate()</i> .....	1382
538	<i>posix_madvise()</i> .....	1384

539	<i>posix_mem_offset()</i> .....	1386
540	<i>posix_memalign()</i> .....	1388
541	<i>posix_spawn()</i> .....	1389
542	<i>posix_spawn_file_actions_addclose()</i> .....	1397
543	<i>posix_spawn_file_actions_adddup2()</i> .....	1400
544	<i>posix_spawn_file_actions_addopen()</i> .....	1402
545	<i>posix_spawn_file_actions_destroy()</i> .....	1403
546	<i>posix_spawn_file_actions_init()</i> .....	1404
547	<i>posix_spawnattr_destroy()</i> .....	1405
548	<i>posix_spawnattr_getflags()</i> .....	1407
549	<i>posix_spawnattr_getpgroup()</i> .....	1409
550	<i>posix_spawnattr_getschedparam()</i> .....	1411
551	<i>posix_spawnattr_getschedpolicy()</i> .....	1413
552	<i>posix_spawnattr_getsigdefault()</i> .....	1415
553	<i>posix_spawnattr_getsigmask()</i> .....	1417
554	<i>posix_spawnattr_init()</i> .....	1419
555	<i>posix_spawnattr_setflags()</i> .....	1420
556	<i>posix_spawnattr_setpgroup()</i> .....	1421
557	<i>posix_spawnattr_setschedparam()</i> .....	1422
558	<i>posix_spawnattr_setschedpolicy()</i> .....	1423
559	<i>posix_spawnattr_setsigdefault()</i> .....	1424
560	<i>posix_spawnattr_setsigmask()</i> .....	1425
561	<i>posix_spawnnp()</i> .....	1426
562	<i>posix_trace_attr_destroy()</i> .....	1427
563	<i>posix_trace_attr_getclockres()</i> .....	1429
564	<i>posix_trace_attr_getinherited()</i> .....	1431
565	<i>posix_trace_attr_getlogsize()</i> .....	1434
566	<i>posix_trace_attr_init()</i> .....	1437
567	<i>posix_trace_clear()</i> .....	1438
568	<i>posix_trace_close()</i> .....	1440
569	<i>posix_trace_create()</i> .....	1442
570	<i>posix_trace_event()</i> .....	1446
571	<i>posix_trace_eventid_equal()</i> .....	1448
572	<i>posix_trace_eventset_add()</i> .....	1450
573	<i>posix_trace_eventtypelist_getnext_id()</i> .....	1452
574	<i>posix_trace_flush()</i> .....	1454
575	<i>posix_trace_get_attr()</i> .....	1455
576	<i>posix_trace_get_filter()</i> .....	1457
577	<i>posix_trace_get_status()</i> .....	1459
578	<i>posix_trace_getnext_event()</i> .....	1460
579	<i>posix_trace_open()</i> .....	1463
580	<i>posix_trace_rewind()</i> .....	1464
581	<i>posix_trace_set_filter()</i> .....	1465
582	<i>posix_trace_shutdown()</i> .....	1466
583	<i>posix_trace_start()</i> .....	1467
584	<i>posix_trace_timedgetnext_event()</i> .....	1469
585	<i>posix_trace_trid_eventid_open()</i> .....	1470
586	<i>posix_trace_trygetnext_event()</i> .....	1471

587	<i>posix_typed_mem_get_info()</i> .....	1472
588	<i>posix_typed_mem_open()</i> .....	1474
589	<i>pow()</i> .....	1477
590	<i>pread()</i> .....	1479
591	<i>printf()</i> .....	1480
592	<i>pselect()</i> .....	1481
593	<i>pthread_atfork()</i> .....	1486
594	<i>pthread_attr_destroy()</i> .....	1488
595	<i>pthread_attr_getdetachstate()</i> .....	1491
596	<i>pthread_attr_getguardsize()</i> .....	1493
597	<i>pthread_attr_getinheritsched()</i> .....	1495
598	<i>pthread_attr_getschedparam()</i> .....	1497
599	<i>pthread_attr_getschedpolicy()</i> .....	1499
600	<i>pthread_attr_getscope()</i> .....	1501
601	<i>pthread_attr_getstack()</i> .....	1503
602	<i>pthread_attr_getstackaddr()</i> .....	1505
603	<i>pthread_attr_getstacksize()</i> .....	1506
604	<i>pthread_attr_init()</i> .....	1507
605	<i>pthread_attr_setdetachstate()</i> .....	1508
606	<i>pthread_attr_setguardsize()</i> .....	1509
607	<i>pthread_attr_setinheritsched()</i> .....	1510
608	<i>pthread_attr_setschedparam()</i> .....	1511
609	<i>pthread_attr_setschedpolicy()</i> .....	1512
610	<i>pthread_attr_setscope()</i> .....	1513
611	<i>pthread_attr_setstack()</i> .....	1514
612	<i>pthread_attr_setstackaddr()</i> .....	1515
613	<i>pthread_attr_setstacksize()</i> .....	1516
614	<i>pthread_barrier_destroy()</i> .....	1517
615	<i>pthread_barrier_init()</i> .....	1519
616	<i>pthread_barrier_wait()</i> .....	1520
617	<i>pthread_barrierattr_destroy()</i> .....	1522
618	<i>pthread_barrierattr_getpshared()</i> .....	1524
619	<i>pthread_barrierattr_init()</i> .....	1526
620	<i>pthread_barrierattr_setpshared()</i> .....	1527
621	<i>pthread_cancel()</i> .....	1528
622	<i>pthread_cleanup_pop()</i> .....	1530
623	<i>pthread_cond_broadcast()</i> .....	1535
624	<i>pthread_cond_destroy()</i> .....	1538
625	<i>pthread_cond_init()</i> .....	1541
626	<i>pthread_cond_signal()</i> .....	1542
627	<i>pthread_cond_timedwait()</i> .....	1543
628	<i>pthread_cond_wait()</i> .....	1548
629	<i>pthread_condattr_destroy()</i> .....	1549
630	<i>pthread_condattr_getclock()</i> .....	1551
631	<i>pthread_condattr_getpshared()</i> .....	1553
632	<i>pthread_condattr_init()</i> .....	1555
633	<i>pthread_condattr_setclock()</i> .....	1556
634	<i>pthread_condattr_setpshared()</i> .....	1557

635	<i>pthread_create()</i> .....	1558
636	<i>pthread_detach()</i> .....	1561
637	<i>pthread_equal()</i> .....	1563
638	<i>pthread_exit()</i> .....	1564
639	<i>pthread_getconcurrency()</i> .....	1566
640	<i>pthread_getcpuclockid()</i> .....	1568
641	<i>pthread_getschedparam()</i> .....	1569
642	<i>pthread_getspecific()</i> .....	1571
643	<i>pthread_join()</i> .....	1573
644	<i>pthread_key_create()</i> .....	1576
645	<i>pthread_key_delete()</i> .....	1580
646	<i>pthread_kill()</i> .....	1582
647	<i>pthread_mutex_destroy()</i> .....	1583
648	<i>pthread_mutex_getprioceiling()</i> .....	1588
649	<i>pthread_mutex_init()</i> .....	1590
650	<i>pthread_mutex_lock()</i> .....	1591
651	<i>pthread_mutex_setprioceiling()</i> .....	1594
652	<i>pthread_mutex_timedlock()</i> .....	1595
653	<i>pthread_mutex_trylock()</i> .....	1597
654	<i>pthread_mutexattr_destroy()</i> .....	1598
655	<i>pthread_mutexattr_getprioceiling()</i> .....	1603
656	<i>pthread_mutexattr_getprotocol()</i> .....	1605
657	<i>pthread_mutexattr_getpshared()</i> .....	1607
658	<i>pthread_mutexattr_gettype()</i> .....	1609
659	<i>pthread_mutexattr_init()</i> .....	1611
660	<i>pthread_mutexattr_setprioceiling()</i> .....	1612
661	<i>pthread_mutexattr_setprotocol()</i> .....	1613
662	<i>pthread_mutexattr_setpshared()</i> .....	1614
663	<i>pthread_mutexattr_settype()</i> .....	1615
664	<i>pthread_once()</i> .....	1616
665	<i>pthread_rwlock_destroy()</i> .....	1618
666	<i>pthread_rwlock_init()</i> .....	1620
667	<i>pthread_rwlock_rdlock()</i> .....	1621
668	<i>pthread_rwlock_timedrdlock()</i> .....	1623
669	<i>pthread_rwlock_timedwrlock()</i> .....	1625
670	<i>pthread_rwlock_tryrdlock()</i> .....	1627
671	<i>pthread_rwlock_trywrlock()</i> .....	1628
672	<i>pthread_rwlock_unlock()</i> .....	1630
673	<i>pthread_rwlock_wrlock()</i> .....	1632
674	<i>pthread_rwlockattr_destroy()</i> .....	1633
675	<i>pthread_rwlockattr_getpshared()</i> .....	1635
676	<i>pthread_rwlockattr_init()</i> .....	1637
677	<i>pthread_rwlockattr_setpshared()</i> .....	1638
678	<i>pthread_self()</i> .....	1639
679	<i>pthread_setcancelstate()</i> .....	1640
680	<i>pthread_setconcurrency()</i> .....	1642
681	<i>pthread_setschedparam()</i> .....	1643
682	<i>pthread_setspecific()</i> .....	1644



## Contents

683	<i>pthread_sigmask()</i> .....	1645
684	<i>pthread_spin_destroy()</i> .....	1647
685	<i>pthread_spin_init()</i> .....	1649
686	<i>pthread_spin_lock()</i> .....	1650
687	<i>pthread_spin_trylock()</i> .....	1652
688	<i>pthread_spin_unlock()</i> .....	1653
689	<i>pthread_testcancel()</i> .....	1654
690	<i>ptsname()</i> .....	1655
691	<i>putc()</i> .....	1656
692	<i>putc_unlocked()</i> .....	1657
693	<i>putchar()</i> .....	1658
694	<i>putchar_unlocked()</i> .....	1659
695	<i>putenv()</i> .....	1660
696	<i>putmsg()</i> .....	1662
697	<i>puts()</i> .....	1666
698	<i>pututxline()</i> .....	1668
699	<i>putwc()</i> .....	1669
700	<i>putwchar()</i> .....	1670
701	<i>pwrite()</i> .....	1671
702	<i>qsort()</i> .....	1672
703	<i>raise()</i> .....	1673
704	<i>rand()</i> .....	1675
705	<i>random()</i> .....	1678
706	<i>read()</i> .....	1679
707	<i>readdir()</i> .....	1687
708	<i>readlink()</i> .....	1691
709	<i>readv()</i> .....	1694
710	<i>realloc()</i> .....	1695
711	<i>realpath()</i> .....	1697
712	<i>recv()</i> .....	1699
713	<i>recvfrom()</i> .....	1701
714	<i>recvmsg()</i> .....	1704
715	<i>regcomp()</i> .....	1707
716	<i>remainder()</i> .....	1715
717	<i>remove()</i> .....	1716
718	<i>remque()</i> .....	1718
719	<i>remquo()</i> .....	1719
720	<i>rename()</i> .....	1721
721	<i>rewind()</i> .....	1725
722	<i>rewinddir()</i> .....	1726
723	<i>rindex()</i> .....	1728
724	<i>rint()</i> .....	1729
725	<i>rmdir()</i> .....	1731
726	<i>round()</i> .....	1735
727	<i>scalb()</i> .....	1736
728	<i>scalbln()</i> .....	1737
729	<i>scanf()</i> .....	1739
730	<i>sched_get_priority_max()</i> .....	1740

731	<i>sched_getparam()</i> .....	1741
732	<i>sched_getscheduler()</i> .....	1742
733	<i>sched_rr_get_interval()</i> .....	1743
734	<i>sched_setparam()</i> .....	1744
735	<i>sched_setscheduler()</i> .....	1747
736	<i>sched_yield()</i> .....	1750
737	<i>seed48()</i> .....	1751
738	<i>seekdir()</i> .....	1752
739	<i>select()</i> .....	1753
740	<i>sem_close()</i> .....	1754
741	<i>sem_destroy()</i> .....	1755
742	<i>sem_getvalue()</i> .....	1756
743	<i>sem_init()</i> .....	1757
744	<i>sem_open()</i> .....	1759
745	<i>sem_post()</i> .....	1762
746	<i>sem_timedwait()</i> .....	1764
747	<i>sem_trywait()</i> .....	1766
748	<i>sem_unlink()</i> .....	1768
749	<i>sem_wait()</i> .....	1770
750	<i>semctl()</i> .....	1771
751	<i>semget()</i> .....	1774
752	<i>semop()</i> .....	1777
753	<i>send()</i> .....	1782
754	<i>sendmsg()</i> .....	1784
755	<i>sendto()</i> .....	1787
756	<i>setbuf()</i> .....	1790
757	<i>setcontext()</i> .....	1791
758	<i>setegid()</i> .....	1792
759	<i>setenv()</i> .....	1793
760	<i>seteuid()</i> .....	1795
761	<i>setgid()</i> .....	1796
762	<i>setgrent()</i> .....	1798
763	<i>sethostent()</i> .....	1799
764	<i>setitimer()</i> .....	1800
765	<i>setjmp()</i> .....	1801
766	<i>setkey()</i> .....	1803
767	<i>setlocale()</i> .....	1805
768	<i>setlogmask()</i> .....	1809
769	<i>setnetent()</i> .....	1810
770	<i>setpgid()</i> .....	1811
771	<i>setpgrp()</i> .....	1814
772	<i>setpriority()</i> .....	1815
773	<i>setprotoent()</i> .....	1816
774	<i>setpwent()</i> .....	1817
775	<i>setregid()</i> .....	1818
776	<i>setreuid()</i> .....	1820
777	<i>setrlimit()</i> .....	1822
778	<i>setservent()</i> .....	1823

## Contents

779	<i>setsid()</i> .....	1824
780	<i>setsockopt()</i> .....	1826
781	<i>setstate()</i> .....	1829
782	<i>setuid()</i> .....	1830
783	<i>setutxent()</i> .....	1833
784	<i>setvbuf()</i> .....	1834
785	<i>shm_open()</i> .....	1836
786	<i>shm_unlink()</i> .....	1840
787	<i>shmat()</i> .....	1842
788	<i>shmctl()</i> .....	1844
789	<i>shmdt()</i> .....	1846
790	<i>shmget()</i> .....	1848
791	<i>shutdown()</i> .....	1850
792	<i>sigaction()</i> .....	1852
793	<i>sigaddset()</i> .....	1859
794	<i>sigaltstack()</i> .....	1860
795	<i>sigdelset()</i> .....	1862
796	<i>sigemptyset()</i> .....	1863
797	<i>sigfillset()</i> .....	1865
798	<i>sighold()</i> .....	1866
799	<i>siginterrupt()</i> .....	1867
800	<i>sigismember()</i> .....	1869
801	<i>siglongjmp()</i> .....	1870
802	<i>signal()</i> .....	1872
803	<i>signbit()</i> .....	1876
804	<i>sigpause()</i> .....	1877
805	<i>sigpending()</i> .....	1878
806	<i>sigprocmask()</i> .....	1879
807	<i>sigqueue()</i> .....	1880
808	<i>sigrelse()</i> .....	1882
809	<i>sigsetjmp()</i> .....	1883
810	<i>sigsuspend()</i> .....	1885
811	<i>sigtimedwait()</i> .....	1887
812	<i>sigwait()</i> .....	1891
813	<i>sigwaitinfo()</i> .....	1893
814	<i>sin()</i> .....	1894
815	<i>sinh()</i> .....	1896
816	<i>sleep()</i> .....	1898
817	<i>socket()</i> .....	1901
818	<i>socketpair()</i> .....	1903
819	<i>sprintf()</i> .....	1905
820	<i>sqrt()</i> .....	1906
821	<i>srand()</i> .....	1908
822	<i>srand48()</i> .....	1909
823	<i>srandom()</i> .....	1910
824	<i>sscanf()</i> .....	1911
825	<i>stat()</i> .....	1912
826	<i>statvfs()</i> .....	1916

827	<i>stdin</i> .....	1917
828	<i>strcasecmp()</i> .....	1919
829	<i>strcat()</i> .....	1920
830	<i>strchr()</i> .....	1921
831	<i>strcmp()</i> .....	1922
832	<i>strcoll()</i> .....	1924
833	<i>strcpy()</i> .....	1926
834	<i>strcspn()</i> .....	1928
835	<i>strdup()</i> .....	1929
836	<i>strerror()</i> .....	1930
837	<i>strfmon()</i> .....	1932
838	<i>strftime()</i> .....	1936
839	<i>strlen()</i> .....	1941
840	<i>strncasecmp()</i> .....	1943
841	<i>strncat()</i> .....	1944
842	<i>strncmp()</i> .....	1945
843	<i>strncpy()</i> .....	1946
844	<i>strpbrk()</i> .....	1947
845	<i>strptime()</i> .....	1948
846	<i>strrchr()</i> .....	1952
847	<i>strspn()</i> .....	1954
848	<i>strstr()</i> .....	1955
849	<i>strtod()</i> .....	1956
850	<i>strtoimax()</i> .....	1960
851	<i>strtok()</i> .....	1961
852	<i>strtoll()</i> .....	1964
853	<i>strtoul()</i> .....	1967
854	<i>strxfrm()</i> .....	1970
855	<i>swab()</i> .....	1972
856	<i>swapcontext()</i> .....	1974
857	<i>swprintf()</i> .....	1975
858	<i>swscanf()</i> .....	1976
859	<i>symlink()</i> .....	1977
860	<i>sync()</i> .....	1979
861	<i>sysconf()</i> .....	1980
862	<i>syslog()</i> .....	1988
863	<i>system()</i> .....	1989
864	<i>tan()</i> .....	1994
865	<i>tanh()</i> .....	1996
866	<i>tcdrain()</i> .....	1998
867	<i>tcflow()</i> .....	2000
868	<i>tcflush()</i> .....	2002
869	<i>tcgetattr()</i> .....	2004
870	<i>tcgetpgrp()</i> .....	2006
871	<i>tcgetsid()</i> .....	2008
872	<i>tcsendbreak()</i> .....	2009
873	<i>tcsetattr()</i> .....	2011
874	<i>tcsetpgrp()</i> .....	2014

## Contents

875	<i>tdelete()</i> .....	2016
876	<i>telldir()</i> .....	2020
877	<i>tempnam()</i> .....	2021
878	<i>tfind()</i> .....	2023
879	<i>tgamma()</i> .....	2024
880	<i>time()</i> .....	2025
881	<i>timer_create()</i> .....	2028
882	<i>timer_delete()</i> .....	2031
883	<i>timer_getoverrun()</i> .....	2032
884	<i>times()</i> .....	2035
885	<i>timezone()</i> .....	2038
886	<i>tmpfile()</i> .....	2039
887	<i>tmpnam()</i> .....	2041
888	<i>toascii()</i> .....	2043
889	<i>tolower()</i> .....	2044
890	<i>toupper()</i> .....	2045
891	<i>towctrans()</i> .....	2046
892	<i>towlower()</i> .....	2047
893	<i>towupper()</i> .....	2048
894	<i>trunc()</i> .....	2049
895	<i>truncate()</i> .....	2050
896	<i>tsearch()</i> .....	2052
897	<i>ttyname()</i> .....	2053
898	<i>twalk()</i> .....	2055
899	<i>tzname</i> .....	2056
900	<i>tzset()</i> .....	2057
901	<i>ualarm()</i> .....	2059
902	<i>ulimit()</i> .....	2061
903	<i>umask()</i> .....	2063
904	<i>uname()</i> .....	2065
905	<i>ungetc()</i> .....	2067
906	<i>ungetwc()</i> .....	2069
907	<i>unlink()</i> .....	2071
908	<i>unlockpt()</i> .....	2076
909	<i>unsetenv()</i> .....	2077
910	<i>usleep()</i> .....	2078
911	<i>utime()</i> .....	2080
912	<i>utimes()</i> .....	2083
913	<i>va_arg()</i> .....	2085
914	<i>vfork()</i> .....	2086
915	<i>vfprintf()</i> .....	2088
916	<i>vfprintf()</i> .....	2090
917	<i>vfwprintf()</i> .....	2091
918	<i>vfwscanf()</i> .....	2092
919	<i>vprintf()</i> .....	2093
920	<i>vscanf()</i> .....	2094
921	<i>vswprintf()</i> .....	2095
922	<i>vswscanf()</i> .....	2096

923	<i>wait()</i> .....	2097
924	<i>waitid()</i> .....	2104
925	<i>waitpid()</i> .....	2106
926	<i>wcrtomb()</i> .....	2107
927	<i>wcscat()</i> .....	2109
928	<i>wcschr()</i> .....	2110
929	<i>wcscmp()</i> .....	2111
930	<i>wcscoll()</i> .....	2112
931	<i>wcscpy()</i> .....	2113
932	<i>wcscspn()</i> .....	2114
933	<i>wcsftime()</i> .....	2115
934	<i>wcslen()</i> .....	2117
935	<i>wcsncat()</i> .....	2118
936	<i>wcsncmp()</i> .....	2119
937	<i>wcsncpy()</i> .....	2120
938	<i>wcsprk()</i> .....	2121
939	<i>wcsrchr()</i> .....	2122
940	<i>wcsrtombs()</i> .....	2123
941	<i>wcsspn()</i> .....	2125
942	<i>wcsstr()</i> .....	2126
943	<i>wcstod()</i> .....	2127
944	<i>wcstoimax()</i> .....	2130
945	<i>wcstok()</i> .....	2131
946	<i>wcstol()</i> .....	2133
947	<i>wcstombs()</i> .....	2136
948	<i>wcstoul()</i> .....	2138
949	<i>wcswcs()</i> .....	2141
950	<i>wcswidth()</i> .....	2142
951	<i>wcsxfrm()</i> .....	2143
952	<i>wctob()</i> .....	2145
953	<i>wctomb()</i> .....	2146
954	<i>wctrans()</i> .....	2148
955	<i>wctype()</i> .....	2149
956	<i>wcwidth()</i> .....	2151
957	<i>wmemchr()</i> .....	2152
958	<i>wmemcmp()</i> .....	2153
959	<i>wmemcpy()</i> .....	2154
960	<i>wmemmove()</i> .....	2155
961	<i>wmemset()</i> .....	2156
962	<i>wordexp()</i> .....	2157
963	<i>wprintf()</i> .....	2162
964	<i>write()</i> .....	2163
965	<i>wscanf()</i> .....	2173
966	<i>y0()</i> .....	2174
967	<b>Index</b> .....	<b>2177</b>

968	<b>List of Tables</b>		
969	2-1	XSI Identifiers .....	519
970	2-2	Sockets Identifiers.....	519
971	2-3	IP Address Resolution Identifiers .....	520
972	2-4	Value of Level for Socket Options.....	566
973	2-5	Socket-Level Options.....	567
974	2-6	Trace Option: System Trace Events.....	580
975	2-7	Trace and Trace Event Filter Options: System Trace Events.....	580
976	2-8	Trace and Trace Log Options: System Trace Events.....	581
977	2-9	Trace, Trace Log, and Trace Event Filter Options: System Trace Events...	581
978	2-10	Trace Option: User Trace Event .....	582





# Preface

980

981 This document is being jointly developed by the IEEE and The Open Group and is intended to  
982 become both IEEE Std. 1003.1-200x and an Open Group Technical Standard, making up the base  
983 volumes of the Single UNIX Specification, Version 3.

## 984 IEEE Std. 1003.1-200x

985 IEEE Std. 1003.1-200x defines the Portable Operating System Interface (POSIX) requirements and  
986 consists of the following volumes:

- 987 • Base Definitions
- 988 • Shell and Utilities
- 989 • System Interfaces (this volume)

## 990 This volume of IEEE Std. 1003.1-200x

991 The System Interfaces volume of IEEE Std. 1003.1-200x describes the interfaces offered to  
992 application programs by POSIX-conformant systems. Readers are expected to be experienced C  
993 language programmers, and to be familiar with the Base Definitions volume of  
994 IEEE Std. 1003.1-200x.

995 This volume of IEEE Std. 1003.1-200x is structured as follows:

- 996 • Chapter 1 explains the status of this volume of IEEE Std. 1003.1-200x and its relationship to  
997 other formal standards.
- 998 • Chapter 2 contains important concepts, terms, and caveats relating to the rest of this volume  
999 of IEEE Std. 1003.1-200x.
- 1000 • Chapter 3 defines the functional interfaces to the POSIX-conformant system.

1001 Comprehensive references are available in the index.

## 1002 Typographical Conventions

1003 The following typographical conventions are used throughout IEEE Std. 1003.1-200x:

- 1004 • **Bold** font is used in text for options to commands, file names, keywords, type names, data  
1005 structures, and their members.
- 1006 • *Italic* strings are used to denote:
  - 1007 — Command operands, command option-arguments, or variable names; for example,  
1008 substitutable argument prototypes
  - 1009 — Environment variables, which are also shown in capitals
  - 1010 — Utility names
  - 1011 — External variables, such as *errno*
  - 1012 — Functions; these are shown as follows: *name()*; names without parentheses are C external  
1013 variables, C function family names, utility names, command operands, or command  
1014 option-arguments.

- 1015 • The font used here is used for the names of constants and literals.
- 1016 • The notation `<file.h>` indicates a header.
- 1017 • Names surrounded by braces, for example, `{ARG_MAX}`, represent symbolic limits or
- 1018 configuration values which may be declared in appropriate headers by means of the C
- 1019 `#define` construct.
- 1020 • The notation `[EABCD]` is used to identify an error value EABCD.
- 1021 • Syntax, code examples, and user input in interactive examples are shown in *fixed width*
- 1022 font. Brackets shown in this font, `[ ]`, are part of the syntax and do *not* indicate optional
- 1023 items. In syntax the `|` symbol is used to separate alternatives, and ellipses (`...`) are used to
- 1024 show that additional arguments are optional.
- 1025 • **Bold fixed width** font is used to identify brackets that surround optional items in syntax,
- 1026 `[ ]`, and to identify system output in interactive examples.
- 1027 • Variables within syntax statements are shown in *italic fixed width* font.
- 1028 • Ranges of values are indicated with parentheses or brackets as follows:
- 1029 — `(a,b)` means the range of all values from `a` to `b`, including neither `a` nor `b`.
- 1030 — `[a,b]` means the range of all values from `a` to `b`, including `a` and `b`.
- 1031 — `[a,b)` means the range of all values from `a` to `b`, including `a`, but not `b`.
- 1032 — `(a,b]` means the range of all values from `a` to `b`, including `b`, but not `a`.
- 1033 • Shading is used to identify extensions; see Section 1.9.1 (on page 501).
- 1034 **Notes:**
- 1035 1. Symbolic limits are used in this volume of IEEE Std. 1003.1-200x instead of
- 1036 fixed values for portability. The values of most of these constants are defined in
- 1037 the Base Definitions volume of IEEE Std. 1003.1-200x, `<limits.h>` or `<unistd.h>`.
- 1038 2. The values of errors are defined in the Base Definitions volume of
- 1039 IEEE Std. 1003.1-200x, `<errno.h>`.

# Trademarks

1040

## 1041 **Notes to Reviewers**

1042 *This section with side shading will not appear in the final copy. - Ed.*

1043 This list will be revised at a later date.

1044 The following information is given for the convenience of users of IEEE Std. 1003.1-200x and  
1045 does not constitute an endorsement by The Open Group or IEEE of these products.

1046 AT&T<sup>®</sup> is a registered trademark of AT&T in the U.S.A. and other countries.

1047 Hewlett-Packard<sup>®</sup>, HP<sup>®</sup>, and HP-UX<sup>®</sup> are registered trademarks of Hewlett-Packard Company.

1048 Motif<sup>®</sup>, OSF/1<sup>®</sup>, UNIX<sup>®</sup>, and the “X Device” are registered trademarks and IT DialTone<sup>™</sup> and  
1049 The Open Group<sup>™</sup> are trademarks of The Open Group in the U.S. and other countries.

1050 POSIX<sup>®</sup> is a registered trademark of the Institute of Electrical and Electronic Engineers, Inc.

1051 /usr/group<sup>®</sup> is a registered trademark of UniForum, the International Network of UNIX System  
1052 Users.

# **Acknowledgements**

## 1054 **Notes to Reviewers**

1055 *This section with side shading will not appear in the final copy. - Ed.*

1056 This list will be revised at a later date.

1057 The Open Group gratefully acknowledges:

- 1058 • AT&T for permission to reproduce portions of its copyrighted System V Interface Definition  
1059 (SVID) and material from the UNIX System V Release 2.0 documentation.
- 1060 • The ANSI X3J11 Committees.
- 1061 • The Large File Summit for their work in developing the set of changes to the X/Open Single  
1062 UNIX Specification to support large files.
- 1063 • The following individuals for their valuable contribution to the development of  
1064 IEEE Std. 1003.1-200x:

1065 TBD

## Referenced Documents

1066

### 1067 **Notes to Reviewers**

1068 *This section with side shading will not appear in the final copy. - Ed.*

1069 This list needs further work, and should be kept in sync with the XBD, Chapter 1, Normative  
1070 References section.

### 1071 **Normative References**

1072 The following standards contain provisions which, through references in IEEE Std. 1003.1-200x,  
1073 constitute provisions of IEEE Std. 1003.1-200x. At the time of publication, the editions indicated  
1074 were valid. All standards are subject to revision, and parties to agreements based on this volume  
1075 of IEEE Std. 1003.1-200x are encouraged to investigate the possibility of applying the most recent  
1076 editions of the standards listed below. Members of IEC and ISO maintain registers of currently  
1077 valid International Standards.

1078 ANS X3.9-1978

1079 (Reaffirmed 1989) American National Standard for Information Systems: Standard  
1080 X3.9-1978, Programming Language FORTRAN.

1081 ISO/IEC 646:1991

1082 ISO/IEC 646:1991, Information Technology — ISO 7-Bit Coded Character Set for  
1083 Information Interchange.

1084 The reference version of the standard contains 95 graphic characters, which are identical to  
1085 the graphic characters defined in the ASCII coded character set.

1086 ISO 4217:1995

1087 ISO 4217:1995, Codes for the Representation of Currencies and Funds.

1088 ISO/IEC 4873:1991

1089 ISO/IEC 4873:1991, Information Technology — ISO 8-Bit Code for Information Interchange  
1090 — Structure and Rules for Implementation.

1091 ISO 8601:1988

1092 ISO 8601:1988, Data Elements and Interchange Formats — Information Interchange —  
1093 Representation of Dates and Times.

1094 ISO/IEC 8859-1:1998

1095 ISO/IEC 8859-1:1998, Information Technology — 8-Bit Single-Byte Coded Graphic  
1096 Character Sets — Part 1: Latin Alphabet No. 1.

1097 This standard character set comprises 191 graphic characters covering the requirements of  
1098 most of Western Europe.

1099 ISO 8859-2:1988

1100 ISO 8859-2:1988, Information Processing — 8-bit Single-byte Coded Graphic Character Sets  
1101 — Part 2: Latin Alphabet No. 2.

1102 ISO C (1999)

1103 ISO/IEC 9899:1999, Programming Languages — C.

1104 ISO POSIX-1:1996

1105 ISO/IEC 9945-1:1996, Information Technology — Portable Operating System Interface

- 1106 (POSIX) — Part 1: System Application Program Interface (API) [C Language] (identical to  
 1107 ANSI/IEEE Std. 1003.1-1996). Incorporating ANSI/IEEE Stds. 1003.1-1990, 1003.1b-1993,  
 1108 1003.1c-1995, and 1003.1i-1995.
- 1109 ISO POSIX-2: 1993  
 1110 ISO/IEC 9945-2: 1993, Information Technology — Portable Operating System Interface  
 1111 (POSIX) — Part 2: Shell and Utilities (identical to IEEE Std. 1003.2-1992 as amended by IEEE  
 1112 Std. 1003.2a-1992).
- 1113 ISO/IEC 10646-1: 1993  
 1114 ISO/IEC 10646-1: 1993, Information Technology — Universal Multiple-Octet Coded  
 1115 Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane.
- 1116 ISO/IEC 14519: 1999  
 1117 ISO/IEC 14519: 1999, Information Technology — POSIX Ada Language Interfaces —  
 1118 Binding for System Application Program Interface (API) — Realtime Extensions.
- 1119 **Informative References**
- 1120 The following documents are referenced in IEEE Std. 1003.1-200x:
- 1121 1984 /usr/group Standard  
 1122 /usr/group Standards Committee, Santa Clara, CA, UniForum 1984.
- 1123 Almasi and Gottlieb  
 1124 George S. Almasi and Allan Gottlieb, *Highly Parallel Computing*, The Benjamin/Cummings  
 1125 Publishing Company, Inc., 1989, ISBN: 0-8053-0177-1.
- 1126 ANSI C  
 1127 American National Standard for Information Systems: Standard X3.159-1989, Programming  
 1128 Language C.
- 1129 ANS X3.226-1994  
 1130 American National Standard for Information Systems: Standard X3.226-1994, Programming  
 1131 Language Common LISP.
- 1132 Brawer  
 1133 Steven Brawer, *Introduction to Parallel Programming*, Academic Press, 1989, ISBN:  
 1134 0-12-128470-0.
- 1135 DeRemer and Pennello Article  
 1136 DeRemer, Frank and Pennello, Thomas J., *Efficient Computation of LALR(1) Look-Ahead Sets*,  
 1137 SigPlan Notices, Volume 15, No. 8, August 1979.
- 1138 Draft ANSI X3J11.1  
 1139 IEEE Floating Point draft report of ANSI X3J11.1 (NCEG).
- 1140 FIPS 151-1  
 1141 Federal Information Procurement Standard (FIPS) 151-1.
- 1142 FIPS 151-2  
 1143 Federal Information Procurement Standards (FIPS) 151-2, Portable Operating System  
 1144 Interface (POSIX)—Part 1: System Application Program Interface (API) [C Language].
- 1145 HP-UX Manual  
 1146 Hewlett-Packard HP-UX Release 9.0 Reference Manual, Third Edition, August 1992.
- 1147 IEC 60559: 1989  
 1148 IEC 60559: 1989, Binary Floating-Point Arithmetic for Microprocessor Systems (previously  
 1149 designated IEC 559: 1989).

## Referenced Documents

- 1150 IEEE Std. 754-1985  
1151 Standard for Binary Floating-Point Arithmetic.
- 1152 IEEE Std. 854-1987  
1153 Standard for Radix-Independent Floating-Point Arithmetic.
- 1154 IEEE Std. 1003.9-1992  
1155 Standard for Information Technology — POSIX FORTRAN 77 Language Interfaces — Part  
1156 1: Binding for System Application Program Interface API.
- 1157 IETF RFC 791  
1158 Internet Protocol, Version 4 (IPv4), September 1981.
- 1159 IETF RFC 819  
1160 The Domain Naming Convention for Internet User Applications, Z. Su, J. Postel, August  
1161 1982.
- 1162 IETF RFC 822  
1163 Standard for the Format of ARPA Internet Text Messages, D.H. Crocker, August 1982.
- 1164 IETF RFC 919  
1165 Broadcasting Internet Datagrams, J. Mogul, October 1984.
- 1166 IETF RFC 920  
1167 Domain Requirements, J. Postel, J. Reynolds, October 1984.
- 1168 IETF RFC 921  
1169 Domain Name System Implementation Schedule, J. Postel, October 1984.
- 1170 IETF RFC 922  
1171 Broadcasting Internet Datagrams in the Presence of Subnets, J. Mogul, October 1984.
- 1172 IETF RFC 1034  
1173 Domain Names — Concepts and Facilities, P. Mockapetris, November 1987.
- 1174 IETF RFC 1035  
1175 Domain Names — Implementation and Specification, P. Mockapetris, November 1987.
- 1176 IETF RFC 1123  
1177 Requirements for Internet Hosts — Application and Support, R. Braden, October 1989.
- 1178 IETF RFC 1886  
1179 DNS Extensions to Support Internet Protocol, Version 6 (IPv6), C. Huitema, S. Thomson,  
1180 December 1995.
- 1181 IETF RFC 2045  
1182 Multipurpose Internet Mail Extensions (MIME), Part 1: Format of Internet Message Bodies,  
1183 N. Freed, N. Borenstein, November 1996.
- 1184 IETF RFC 2373  
1185 Internet Protocol, Version 6 (IPv6) Addressing Architecture, S. Deering, R. Hinden, July  
1186 1998.
- 1187 IETF RFC 2460  
1188 Internet Protocol, Version 6 (IPv6), S. Deering, R. Hinden, December 1998.
- 1189 Internationalisation Guide  
1190 Guide, July 1993, Internationalisation Guide, Version 2 (ISBN: 1-859120-02-4, G304),  
1191 published by The Open Group.

- 1192 ISO C (1990)  
 1193 ISO/IEC 9899:1990: Programming Languages — C, including Amendment 1:1995 (E), C  
 1194 Integrity (Multibyte Support Extensions (MSE) for ISO C).
- 1195 ISO 2375:1985  
 1196 ISO 2375:1985, Data Processing — Procedure for Registration of Escape Sequences.
- 1197 ISO/IEC 1539:1990  
 1198 ISO/IEC 1539:1990, Information Technology — Programming Languages — Fortran  
 1199 (technically identical to the ANSI X3.9-1978 standard [FORTRAN 77]).
- 1200 ISO/IEC 6429:1992  
 1201 ISO/IEC 6429:1992, Information Technology — Control Functions for Coded Character  
 1202 Sets.
- 1203 ISO/IEC 6937:1994  
 1204 ISO/IEC 6937:1994, Information Technology — Coded Character Set for Text  
 1205 Communication — Latin Alphabet.
- 1206 ISO 7-bit or 8-bit coded character set for text communication using public communication  
 1207 networks, private communication networks, or interchange media, such as magnetic tapes  
 1208 and discs.
- 1209 ISO/IEC 8802-3:1996  
 1210 ISO/IEC 8802-3:1996, Information Technology — Telecommunications and Information  
 1211 Exchange Between Systems — Local and Metropolitan Area Networks — Specific  
 1212 Requirements — Part 3: Carrier Sense Multiple Access with Collision Detection  
 1213 (CSMA/CD) Access Method and Physical Layer Specifications.
- 1214 Issue 1  
 1215 X/Open Portability Guide, July 1985 (ISBN: 0-444-87839-4).
- 1216 Issue 2  
 1217 X/Open Portability Guide, January 1987:
- 1218 • Volume 1: XVS Commands and Utilities (ISBN: 0-444-70174-5)
  - 1219 • Volume 2: XVS System Calls and Libraries (ISBN: 0-444-70175-3)
- 1220 Issue 3  
 1221 X/Open Specification, 1988, 1989, February 1992:
- 1222 • Commands and Utilities, Issue 3 (ISBN: 1-872630-36-7, C211); this specification was  
 1223 formerly X/Open Portability Guide, Issue 3, Volume 1, January 1989, XSI Commands  
 1224 and Utilities (ISBN: 0-13-685835-X, XO/XPG/89/002)
  - 1225 • System Interfaces and Headers, Issue 3 (ISBN: 1-872630-37-5, C212); this specification  
 1226 was formerly X/Open Portability Guide, Issue 3, Volume 2, January 1989, XSI System  
 1227 Interface and Headers (ISBN: 0-13-685843-0, XO/XPG/89/003)
  - 1228 • Curses Interface, Issue 3, contained in Supplementary Definitions, Issue 3  
 1229 (ISBN: 1-872630-38-3, C213), Chapters 9 to 14 inclusive; this specification was formerly  
 1230 X/Open Portability Guide, Issue 3, Volume 3, January 1989, XSI Supplementary  
 1231 Definitions (ISBN: 0-13-685850-3, XO/XPG/89/004)
  - 1232 • Headers Interface, Issue 3, contained in Supplementary Definitions, Issue 3  
 1233 (ISBN: 1-872630-38-3, C213), Chapter 19, Cpio and Tar Headers; this specification was  
 1234 formerly X/Open Portability Guide Issue 3, Volume 3, January 1989, XSI Supplementary  
 1235 Definitions (ISBN: 0-13-685850-3, XO/XPG/89/004)



## Referenced Documents

- 1236 Issue 4  
1237 CAE Specification, July 1992, published by The Open Group:
- 1238 • System Interface Definitions (XBD), Issue 4 (ISBN: 1-872630-46-4, C204)
  - 1239 • Commands and Utilities (XCU), Issue 4 (ISBN: 1-872630-48-0, C203)
  - 1240 • System Interfaces and Headers (XSH), Issue 4 (ISBN: 1-872630-47-2, C202)
- 1241 Issue 4, Version 2  
1242 CAE Specification, August 1994, published by The Open Group:
- 1243 • System Interface Definitions (XBD), Issue 4, Version 2 (ISBN: 1-85912-036-9, C434)
  - 1244 • Commands and Utilities (XCU), Issue 4, Version 2 (ISBN: 1-85912-034-2, C436)
  - 1245 • System Interfaces and Headers (XSH), Issue 4, Version 2 (ISBN: 1-85912-037-7, C435)
- 1246 Issue 5  
1247 Technical Standard, February 1997, published by The Open Group:
- 1248 • System Interface Definitions (XBD), Issue 5 (ISBN: 1-85912-186-1, C605)
  - 1249 • Commands and Utilities (XCU), Issue 5 (ISBN: 1-85912-191-8, C604)
  - 1250 • System Interfaces and Headers (XSH), Issue 5 (ISBN: 1-85912-181-0, C606)
- 1251 Knuth Article  
1252 Knuth, Donald E., *On the Translation of Languages from Left to Right*, Information and Control,  
1253 Volume 8, No. 6, October 1965.
- 1254 KornShell  
1255 Bolsky, Morris I. and Korn, David G., *The New KornShell Command and Programming*  
1256 *Language*, March 1995, Prentice Hall.
- 1257 MSE working draft  
1258 Working draft of ISO/IEC 9899:1990/Add3:draft, Addendum 3 — Multibyte Support  
1259 Extensions (MSE) as documented in the ISO Working Paper SC22/WG14/N205 dated 31  
1260 March 1992.
- 1261 POSIX.1: 1988  
1262 IEEE Std. 1003.1-1988, Standard for Information Technology — Portable Operating System  
1263 Interface (POSIX) — Part 1: System Application Program Interface (API) [C Language].
- 1264 POSIX.1: 1990  
1265 IEEE Std. 1003.1-1990, Standard for Information Technology — Portable Operating System  
1266 Interface (POSIX) — Part 1: System Application Program Interface (API) [C Language].
- 1267 POSIX.1a: 2000  
1268 IEEE Std. 1003.1a-2000, Standard for Information Technology — Portable Operating System  
1269 Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment  
1270 ????: TITLE?? [C Language].
- 1271 POSIX.1d: 1999  
1272 IEEE Std. 1003.1d-1999, Standard for Information Technology — Portable Operating System  
1273 Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment  
1274 ????: Additional Realtime Extensions [C Language].
- 1275 POSIX.1g: 2000  
1276 IEEE Std. 1003.1g-2000, Standard for Information Technology — Portable Operating System  
1277 Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment  
1278 ????: Protocol-Independent Interfaces (PII).

- 1279 POSIX.1j: 2000  
 1280 IEEE Std. 1003.1j-2000, Standard for Information Technology — Portable Operating System  
 1281 Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment  
 1282 ??: Advanced Realtime Extensions [C Language].
- 1283 POSIX.1q: 2000  
 1284 IEEE Std. 1003.1q-2000, Standard for Information Technology — Portable Operating System  
 1285 Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment  
 1286 ??: Tracing [C Language].
- 1287 POSIX.2: 1992  
 1288 IEEE Std. 1003.2-1992, Standard for Information Technology — Portable Operating System  
 1289 Interface (POSIX) — Part 2: Shell and Utilities.
- 1290 POSIX.2b:-2000  
 1291 IEEE Std. 1003.2b: 2000, Standard for Information Technology — Portable Operating System  
 1292 Interface (POSIX) — Part 2: Shell and Utilities — Amendment ??: TITLE??.
- 1293 POSIX.2d:-1994  
 1294 IEEE Std. 1003.2d: 1994, Standard for Information Technology — Portable Operating System  
 1295 Interface (POSIX) — Part 2: Shell and Utilities — Amendment 1: Batch Environment.
- 1296 Sarwate Article  
 1297 Sarwate, Dilip V., *Computation of Cyclic Redundancy Checks via Table Lookup*, Communications  
 1298 of the ACM, Volume 30, No. 8, August 1988.
- 1299 SVID, Issue 1  
 1300 American Telephone and Telegraph Company, System V Interface Definition (SVID), Issue  
 1301 1; Morristown, NJ, UNIX Press, 1985.
- 1302 SVID, Issue 2  
 1303 American Telephone and Telegraph Company, System V Interface Definition (SVID), Issue  
 1304 2; Morristown, NJ, UNIX Press, 1986.
- 1305 SVID, Issue 3  
 1306 American Telephone and Telegraph Company, System V Interface Definition (SVID), Issue  
 1307 3; Morristown, NJ, UNIX Press, 1989.
- 1308 The AWK Programming Language  
 1309 Aho, Alfred V., Kernighan, Brian W., and Weinberger, Peter J., *The AWK Programming*  
 1310 *Language*, Reading, MA, Addison-Wesley 1988.
- 1311 XNS, Issue 4  
 1312 CAE Specification, August 1994, Networking Services, Issue 4 (ISBN: 1-85912-049-0, C438),  
 1313 published by The Open Group.
- 1314 XNS, Issue 5  
 1315 CAE Specification, February 1997, Networking Services, Issue 5 (ISBN: 1-85912-165-9, C523),  
 1316 published by The Open Group.
- 1317 XNS, Issue 5.2  
 1318 Technical Standard, January 2000, Networking Services (XNS), Issue 5.2  
 1319 (ISBN: 1-85912-241-8, C808), published by The Open Group.
- 1320 X/Open Curses, Issue 4, Version 2  
 1321 CAE Specification, May 1996, X/Open Curses, Issue 4, Version 2 (ISBN: 1-85912-171-3,  
 1322 C610), published by The Open Group.

## Referenced Documents

- 1323 UTF-8  
1324 ISO/IEC 10646-1: 1993/Amendment 2: 1996, UCS Transformation Format 8 (UTF-8).
- 1325 Yacc: Yet Another Compiler Compiler  
1326 REFERENCE NEEDED.
- 1327 Parts of the following documents were used to create the base documents for  
1328 IEEE Std. 1003.1-200x:
- 1329 AIX 3.2 Manual  
1330 AIX Version 3.2 For RISC System/6000, Technical Reference: Base Operating System And  
1331 Extensions, 1990, 1992 (Part No. SC23-2382-00).
- 1332 OSF/1  
1333 OSF/1 Programmer's Reference, Release 1.2 (ISBN: 0-13-020579-6).
- 1334 OSF AES  
1335 Application Environment Specification (AES) Operating System Programming Interfaces  
1336 Volume, Revision A (ISBN: 0-13-043522-8).
- 1337 System V Release 2.0  
1338 — UNIX System V Release 2.0 Programmer's Reference Manual (April 1984 - Issue 2).  
1339 — UNIX System V Release 2.0 Programming Guide (April 1984 - Issue 2).
- 1340 System V Release 4.2  
1341 Operating System API Reference, UNIX® SVR4.2 (1992) (ISBN: 0-13-017658-3).

